

pom's

La revue francophone des utilisateurs de l'Apple

- 🍏 Compte bancaire sur Macintosh
- 🍏 Utilisation de la carte langage
- 🍏 Deux fonctions pour ProDOS
- 🍏 DOS et ProDOS en mémoire
- 🍏 Copie d'écran sur Silentype
- 🍏 Manipulation de catalogue
- 🍏 Assembleur et ProDOS
- 🍏 Zoom HGR
- 🍏 Fredisk



NUMERO 20 • PRIX 40 F

ISSN 0294-6068

APPLE CHEZ P.S.I.



Apple, logique et systèmes experts,
par René DESCAMPS.
224 pages - 120,00 FF.
12 programmes en Basic, 4 fichiers proposent de construire un "moteur d'inférence" capable de trouver la solution d'énigmes et de bâtir des raisonnements.

Basic Plus 80 routines sur Apple II,
par Michel MARTIN.
144 pages - 95,00 FF.
Basic Plus vous propose 80 routines pour "muscler" votre machine, 80 manières de simuler des fonctions que vous n'auriez jamais cru pouvoir utiliser.

Exploitation d'enquêtes sur Apple II et IBM/PC,
par J.-F. GRIMMER.
176 pages - 120,00 FF.
L'ensemble des programmes proposés permet d'ex-

ploiter, avec quelques connaissances informatiques, des fichiers de données numériques sur Apple II et IBM/PC.

Diététique sur Apple II,
224 pages - 120,00 FF.
Mangez équilibré grâce à votre ordinateur.

Apple en famille,
par J.-F. SEHAN.
232 pages - 120,00 FF.
40 programmes utiles en Basic.

Super Jeux Apple,
256 pages - 120,00 FF.
50 programmes de jeux en Basic sur Apple.

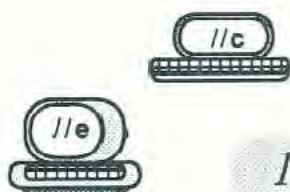
Apple, modems et serveurs,
par Alain MARIATTE.
224 pages - 130,00 FF.
Pour apprendre à se servir d'un modem et se familiariser avec les outils télématiques.



P.S.I. DIFFUSION B.P. 86
77402 LAGNY-SUR-MARNE CEDEX
Tél.: (6) 006.44.35
Télex: PSIDIF 600978 F

Dos ou ProDOS à la carte

François Sermier



16

Des messages en boîtes

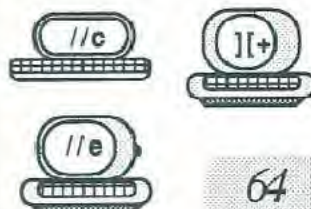
Gérard Michel



41

Décalages...

Pascal Cantot



64

Manipulation de catalogue

Jean-Luc Arnaud



23

Gestion de comptes bancaires

Jean-Luc Bazanegue



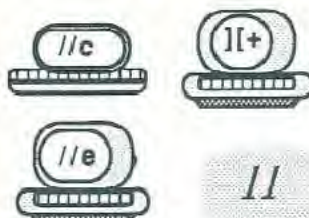
44

Les annonceurs

P.S.I.	2
M.B.D.C.	15
APPLE	38/39
Ordinateur Individuel	40
Télécompo	63
Club Apple	75
Guide-Micro O.J.	76

Utilisation de la carte langage

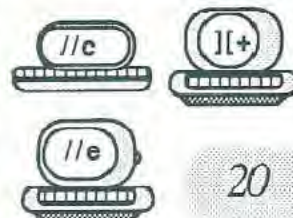
Hervé Roy-Contancin



11

Copie d'écran graphique sur Silentype

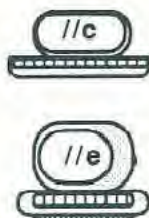
Gérard Becher



20

Allô ProDOS ?

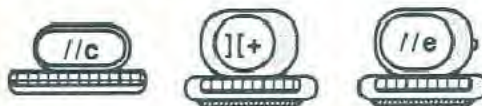
Alexandre Avrane



50

Quatre fonctions pour le Basic

Sylvie Gallet



5

Editorial

Hervé Thiriez



4

Courrier des Lecteurs

Olivier Herz



71

Micro-infos

Jean-Michel Gourévitch



67

100 %

laser et 100 % Apple pour la préparation de Pom's à partir de ce numéro. Contribuent aux pages qui suivent : des Macs 128 et 512 Ko, une imprimante à laser, des Apple][+, IIe, IIc, des liaisons série pour que ce beau monde communique, MacPaint, MacDraw, MacWrite, AppleWriter et quelques programmes 'maison'. Cela pour une présentation que vous trouverez, nous l'espérons, améliorée et qui prendra certainement son aspect définitif dès le numéro 21.

Les congés vous ont permis, si ce n'est de bronzer, tout au moins de penser à Pom's. Nous recevons de nombreuses contributions qui nous promettent de belles pages pour les prochains numéros : continuez !

Toujours un sommaire fourni :

De nouvelles fonctions accessibles par l'& après un simple BRUN : tel est l'objet du programme de **Sylvie Gallet**. Une utilisation originale de votre carte langage - y stocker des variables chaînes de longueur fixe - il s'agit d'un utilitaire de **Hervé Roy-Contancin**.

Deux articles consacrés à ProDOS dans ce numéro :

- une très belle de contribution de **François Sermier** qui permet de disposer de DOS et ProDOS simultanément en mémoire. Pourquoi ne pas éditer vos programmes Basic avec EPE sous DOS puis les sauvegarder sous ProDOS ?
- une étude approfondie des appels en assembleur par **Alexandre Avrane** avec la mise en place de commandes externes, un Tdump et un Init qui faisaient défaut.

Gérard Becher a écrit un programme de copie d'écran sur Silentyte qui, outre son aspect pratique pour les possesseurs de ces imprimantes, présente un intérêt didactique pour tous par son source particulièrement commenté. Dans le même esprit, avec **Pascal Cantot**, les Asl, Lsr, Rol et Ror ne devraient plus avoir de secret pour vous.

Egalement, un volumineux programme assembleur de **Jean-Luc Arnaud** pour éditer le catalogue de vos disquettes.

Converti au 68000, nous retrouvons **Gérard Michel** dans le cahier Mac. Belle contribution de **Jean-Luc Bazanegue** et **Christian Piard** : La gestion de vos comptes bancaires personnels sur Macintosh, présentation "Macintosh" bien qu'exceptionnellement sans coccinelle !...

Et, comme à l'accoutumée, les rubriques de **Jean-Michel Gourévitch** et **Olivier Herz**.

Hervé Thiriez

Ont collaboré à ce numéro : Alexandre Avrane, Jean-Luc Arnaud, Jean-Luc Bazanegue, Gérard Becher, Pascal Cantot, Sylvie Gallet, Jean-Michel Gourévitch, Olivier Herz, Gérard Michel, Hervé Roy-Contancin, François Sermier.

Directeur de la publication, rédacteur en chef : Hervé Thiriez.

Rédacteurs : Alexandre Avrane, Olivier Herz.

Siège social : Editions MEV, 49, rue Lamartine, 78000 Versailles. Tél : (3) 951 24 43.

Publicité : Editions MEV.

Diffusion : N. M. P. P.

Impression : Rosay, 47, avenue de Paris, 94300 Vincennes. Tél : (1) 328 18 63.

Quatre fonctions pour le Basic

Sylvie Gallet

Le programme présenté ici est classique, mais est d'une grande utilité pour les programmeurs en Basic. En effet, par l'intermédiaire de l'ampersand, il permet :

- de supprimer les REMs d'un programme Basic. Toutefois, il ne tient pas compte des GOTOs et des GOSUBs ;
- d'éditer une ligne de programme, sans introduire d'espace superflu. Un petit problème subsiste : il s'agit des REMs comportant un CTRL-J ;
- le POKE 33,28 (bien pratique) ;
- l'accès au Mini-Assembleur ; celui-ci a été relogé en \$8A9A mais peut être aussi appelé par \$8C00G ;
- l'accès à toutes les fonctions du RENUMBER de la disquette Maître.

Mode d'emploi

En lançant le programme COMMANDES, vous obtiendrez la liste des commandes autorisées ainsi que leur syntaxe. Le programme UTILITAIRES contient déjà RENUMBER et le Mini-Assembleur. Il ajuste lui-même la HIMEM et le vecteur ampersand (&). Il se lance par :

BRUN UTILITAIRES, A\$8A8E
Il y a une petite précaution à prendre. Après MAXFILES, HIMEM ou FP, il est indispensable de relancer UTILITAIRES.

Les commandes

La syntaxe

- &P effectue successivement un HOME et un POKE 33,28
- &LNNN liste la ligne de programme dont le numéro est donné par NNN
- &C supprime les REMarques du programme
- &A donne accès au Mini-Assembleur (Basic)
- 8C00G pour le Mini-Assembleur (Moniteur)

Pour le RENUMBER, la syntaxe est un peu plus lourde. Il faudra taper : "RENUMBER:&" suivi des paramètres F, I, S, E, H, M. Ces paramètres représentent respectivement :

- le premier nouveau numéro de ligne (first);
- l'incrément ;
- le numéro de la première ligne à renuméroter (start);
- le numéro de la dernière ligne à renuméroter (end);
- la protection du programme pour fusion (hold);
- la réalisation de la fusion

(merge);

Pour les utilisateurs n'ayant pas la disquette d'accompagnement, voilà comment il faut procéder pour RENUM et MINI :

```
] FP (HIMEM: 38400 soit
$9600 en hexa)
] RUN RENUMBER
]BSAVE RENUM, A$8E00,
L$800
] BLOAD INTBASIC, A$6000
] CALL-151
* 8535: 4C 2F 8B
* 8559: 4C 2F 8B
* 85BD: 20 CE 8B
* 85DB: 20 CE 8B
* 85E5: 20 CE 8B
* 8631: 4C F6 8A
* 8666: 4C 2C 8B
* BSAVE MINI, A$8500,
L$170
* BLOAD MINI, A$8A9A
* BSAVE MINI, A$8A9A,
L$170
* BLOAD UTILITAIRES OBJ,
A$8A8E
* BLOAD MINI
* BLOAD RENUM
* BSAVE UTILITAIRES,
A$8A8E, L$B72
```



Programme UTILITAIRES SCE (Assembleur LISA 2.5)

```
1 ;
2 ;
3 *****
4 *
5 * UTILITAIRES DE BASIC *
6 *
7 * S. GALLET 27-2-1985 *
```

```
8 *
9 *****
10 ;
11 ;
12 STXERR EQU $DEC9
13 HOME EQU $FC58
14 FRMNUM EQU $DD67
15 GETADR EQU $E752
16 FNDLIN EQU $D61A
17 ILLQTY EQU $E199
18 OUTDO EQU $DB5C
19 GIVAYF EQU $E2F2
20 MOVAF EQU $EB63
```

```
21 FDI VT EQU $EA69
22 CONINT EQU $E6FB
23 LINPRNT EQU $ED24
24 CRDO EQU $DAFB
25 BASIC EQU $FEB3
26 SIGNFLT EQU $E301
27 AMPER EQU $3F5
28 RENUMBER EQU $8E00
29 MINI EQU $8C00
30 ;
31 LARGFEN EPZ $21
32 CHRGET EPZ $B1
```


33	LOWTR	EPZ \$9B	103	JMP MINI	173	BPL PRINT
34	LINNUM	EPZ \$50	104	T5:	174	TOKEN:
35	AUXIL	EPZ \$85	105	JMP RENUMBER	175	SEC
36	FAC	EPZ \$9D	106	ERR:	176	SBC #\$7F
37	CV	EPZ \$25	107	JMP STXERR	177	TAX
38	CH	EPZ \$24	108	;	178	STY AUXIL
39	LT	EPZ \$FE	109	;	179	LDY #\$D0
40	TXTPTR	EPZ \$B8	110	POKE:	180	STY FAC
41	AUX	EPZ \$FC	111	;----	181	LDY #\$CF
42	HIMEM	EPZ \$73	112	JSR CHRGET	182	STY FAC+1
43	CAR.LU	EPZ \$FA	113	BNE ERR	183	LDY #\$FF
44	CAR.ECR	EPZ \$FC	114	JSR HOME	184	LIT:
45	LS	EPZ \$FE	115	LDA #\$1C	185	DEX
46	TEMP	EPZ \$F9	116	STA LARGFEN	186	BEQ BONTOKEN
47	PAGEND	EPZ \$AF	117	RTS	187	LIT1:
48	TXTTAB	EPZ \$67	118	;	188	JSR INCR.Y1
49	;		119	;	189	BPL LIT1
50	;		120	EDIT:	190	BMI LIT
51		ORG \$8A8E	121	;----	191	BONTOKEN:
52		OBJ \$800	122	LDA TXTPTR	192	LDA #\$20
53	;		123	STA AUX	193	JSR OUTDO1
54	*****		124	LDA TXTPTR+1	194	PRNTINST:
55	;		125	STA AUX+1	195	JSR INCR.Y1
56	ORIGINE:		126	LITCAR:	196	BMI DERNCAR
57	;-----		127	JSR CHRGET	197	JSR OUTDO1
58	NOP		128	BCC LITCAR	198	BNE PRNTINST
59	LDA #ORIGINE		129	BNE ERR	199	DERNCAR:
60	STA HIMEM		130	LDA AUX	200	JSR OUTDO1
61	LDA /ORIGINE		131	STA TXTPTR	201	LDY AUXIL
62	STA HIMEM+1		132	LDA AUX+1	202	LDA #\$20
63	JMP INIT		133	STA TXTPTR+1	203	PRINT:
64	;		134	INC TXTPTR	204	JSR OUTDO1
65	*****		135	BNE CALCFORM	205	JMP BOUCLE
66	;		136	INC TXTPTR+1	206	FIN:
67	;DEBUT DU MINI ASSEMBLEUR		137	CALCFORM:	207	LDA LT+1
68	;		138	JSR FRMNUM	208	LDY LT
69	;FIN DU MINI ASSEMBLEUR		139	BCS ERR	209	JSR CIVAYF
70	MINIASD:		140	JSR GETADR	210	JSR MOVAF
71	;		141	LDA #\$F9	211	LDY LARGFEN
72	*****		142	CMP LINNUM+1	212	JSR SIGNFLT
73	;		143	BCS CHLIGN	213	JSR FDIIVT
74		ORG MINIASD:\$170	144	JMP ILLQTY	214	JSR CONTINT
75		OBJ \$97C	145	CHLIGN:	215	TXA
76	;		146	JSR FNDLIN	216	EOR #\$FF
77	INIT:		147	BCS PRTLIGN	217	CLC
78	;----		148	JMP STOP	218	ADC CV
79		LDA #\$4C	149	PRTLIGN:	219	BMI STOP
80		STA AMPER	150	JSR CRDO	220	CMP #\$18
81		LDA #DEBUT	151	JSR CRDO	221	BPL STOP
82		STA AMPER+1	152	LDA #\$20	222	STA CV
83		LDA /DEBUT	153	JSR OUTDO1	223	STOP:
84		STA AMPER+2	154	LDA #\$20	224	JMP BASIC
85		JMP BASIC	155	JSR OUTDO1	225	;
86	DEBUT:		156	LDA LINNUM+1	226	;
87	;-----		157	LDX LINNUM	227	*** SOUS-PROGRAMMES ***
88	T1:		158	JSR LINPRNT	228	;
89		CMP #'P'	159	LDA CH	229	;
90		BNE T2	160	STA LT	230	OUTDO1:
91		JMP POKE	161	LDA #\$00	231	;-----
92	T2:		162	STA LT+1	232	JSR OUTDO
93		CMP #'L'	163	LDA #\$20	233	INC LT
94		BNE T3	164	JSR OUTDO1	234	BNE RETURN
95		JMP EDIT	165	LIST:	235	INC LT+1
96	T3:		166	LDY #\$3	236	RETURN:
97		CMP #'C'	167	BOUCLE:	237	RTS
98		BNE T4	168	JSR INCR.Y2	238	;
99		JMP COMPACTE	169	LDA (LOWTR),Y	239	;
100	T4:		170	BNE SUITE	240	INCR.Y1:
101		CMP #'A'	171	JMP FIN	241	;-----
102		BNE T5	172	SUITE:	242	INY

243	BNE MPAG1	289	ORA TEMP	335	RTS
244	INC FAC+1	290	BEQ FIN.REM	336 ;	
245	MPAG1:	291	LDX #52	337 ECRIRE	INC CAR.ECR
246	LDA (FAC),Y	292 BX2	JSR LIRE	338	BNE L
247	RTS	293	JSR ECRIRE	339	INC CAR.ECR+1
248 ;		294	DEX	340 E	STA (CAR.ECR),Y
249 ;		295	BNE BX2	341	RTS
250 INCR.Y2:		296	JMP DEBUTREM	342 ;	
251 ;-----		297 FIN.REM	JSR ECRIRE	343 LIRE	INC CAR.LU
252	INY	298	JSR ECRIRE	344	BNE L
253	BNE MPAG2	299	LDA CAR.ECR	345	INC CAR.LU+1
254	INC LOWTR+1	300	STA PAGEND	346 L	LDA CAR.LU+1
255 MPAG2:		301	LDA CAR.ECR+1	347	CMP PAGEND+1
256	RTS	302	STA PAGEND+1	348	BNE LL
257 ;		303	JMP BASIC	349	LDA CAR.LU
258 ;		304 SUITEREM	CMP #5B2	350	CMP PAGEND
259 *****		305	BEQ REM	351	BNE LL
260 ;		306	JSR ECRIRE	352	JMP FIN.REM
261 ;		307	JMP DEBUTREM	353 LL	LDA (CAR.LU),Y
262 ;		308 REM	LDA #53	354	RTS
263 COMPACTE:		309	CLC	355 ;	
264 ;-----		310	ADC LS	356 CHAINER	LDA CAR.ECR
265 ;		311	PHP	357	CLC
266 INIT.REM LDY #50		312	CMP CAR.ECR	358	ADC #51
267	LDA TXTTAB	313	BNE NON	359	STA (LS),Y
268	STA LS	314	LDA #50	360	LDA CAR.ECR+1
269	CLC	315	PLP	361	ADC #50
270	ADC #53	316	ADC LS+1	362	INY
271	STA CAR.LU	317	CMP CAR.ECR+1	363	STA (LS),Y
272	STA CAR.ECR	318	BNE NON	364	DEY
273	LDA TXTTAB+1	319 REM.FL	LDA #54	365	STA LS+1
274	STA LS+1	320	STA TEMP	366	LDA CAR.ECR
275	ADC #50	321	JSR DECALE	367	STA LS
276	STA CAR.LU+1	322	JMP TESTFIN	368	INC LS
277	STA CAR.ECR+1	323 NON	LDA #51	369	RTS
278 ;		324	STA TEMP	370 ;	
279 DEBUTREM JSR LIRE		325	JSR DECALE	371 *****	
280	BNE SUITEREM	326	JMP ZFL	372 ;	
281 ZFL JSR ECRIRE		327 DECALE	LDA CAR.ECR	373 ;	RENUMBER SE TROUVE ICI
282	JSR CHAINER	328	SEC	374 ;	
283 TESTFIN LDX #52		329	SBC TEMP	375 *****	
284 BX1 STA TEMP		330	BCS X	376 ;	
285 JSR LIRE		331	DEC CAR.ECR+1	377 FINAL:	
286 JSR ECRIRE		332 X	STA CAR.ECR	378 ;	
287 DEX		333 BOUCLE.L	JSR LIRE	379 LONGUEUR EQU FINAL-ORIGINE	
288 BNE BX1		334	BNE BOUCLE.L	380	END

Programme COMMANDES

```

10 TEXT:HOME: SPEED=255:D$= CHR$(4)
20 PRINT "      COMMANDES:"
30 PRINT:PRINT"-8C00G  MINI ASSEMBLEUR
   (MONITEUR)":PRINT"-&A  MINI ASSEMBLEUR
   (BASIC)"
40 PRINT:PRINT"-&P  HOME+POKE33,28":PRINT:
   PRINT"-&LNNN  LISTE LA LIGNE NNN POUR":
   PRINT"      EDITION"
50 PRINT:PRINT"-&C  SUPPRIME LES
   REMARQUES"
60 PRINT:PRINT"RENUMBER: & SUIVI DE:":PRINT"-F

```

```

1ER NOUVEAU NUMERO"
70 PRINT"-I  INCREMENT":PRINT"-S  1ERE
   LIGNE A RENUMEROTER"
80 PRINT"-E  DERNIERE LIGNE A RENUMEROTER"
90 PRINT"-H  PROTEGE LE PRGM POUR FUSION":
   PRINT"-M  REALISE LA FUSION"
100 PRINT:PRINT"ATTENTION! APRES 'MAXFILES',
   'HIMEM' OU 'FP', IL FAUT RELANCER
   'UTILITAIRES'"
110 PRINT D$"BRUNUTILITAIRES"

```


Récapitulation UTILITAIRES OBJ

8A8E- EA A9
8A90- 8E 85 73 A9 8A 85 74 4C
8A98- 0A 8C A5 8D 9F A5 A9 9D
8AA0- 8D 7C A5 8D A0 A5 A9 EA
8AA8- 8D DF A6 8D E0 A6 20 58
8AB0- FC A2 00 BD 30 08 F0 17
8AB8- 20 ED FD E8 D0 F5 8D CC
8AC0- CF C1 C4 C9 CE C7 A0 CC
8AC8- C9 D3 C1 AE AE AE 00 AD
8AD0- 83 C0 AD 83 C0 A9 00 8D
8AD8- 00 E0 AD 00 E0 D0 70 A9
8AE0- FF 8D 00 E0 AD 00 E0 C9
8AE8- FF D0 64 A0 00 AD 89 C0
8AF0- AD 89 C0 B9 8B 08 F0 07
8AF8- 20 ED FD C8 4C 65 08 AD
8B00- 83 C0 AD 83 C0 AD 83 C0
8B08- A0 00 B9 A6 08 F0 07 20
8B10- ED FD C8 4C 7C 08 4C 00
8B18- E0 8D 84 C2 CC CF C1 C4
8B20- A0 C1 D3 CD AE C3 CF C4
8B28- C5 AE B1 AC C1 A4 C4 B0
8B30- B0 B0 8D 00 8D 84 C2 CC
8B38- CF C1 C4 A0 C1 D3 CD AE
8B40- C3 CF C4 C5 AE B2 AC C1
8B48- A4 C4 B0 B0 B0 8D 00 A9
8B50- 01 8D B1 AA A2 00 BD D6
8B58- 08 F0 06 20 ED FD E8 D0
8B60- F5 4C 00 60 8D 84 CD C1
8B68- D8 C6 C9 CC C5 D3 A0 B1
8B70- 8D 84 C2 CC CF C1 C4 A0
8B78- C1 D3 CD AE C3 CF C4 C5
8B80- AC C1 A4 B6 B0 B0 B0 8D
8B88- 00 00 00 00 B7 FF FF FF
8B90- 61 01 FE FE 01 01 FF FF
8B98- 01 01 FE FF 01 01 FF FF
8BA0- E1 01 FE FE 01 01 FF FF
8BA8- 01 01 FE FF 01 01 FE FF
8BB0- 21 01 FE FE 01 00 FF FF
8BB8- 00 00 FE FF 01 00 FE FF
8BC0- 41 01 FE FE 01 00 FF FF
8BC8- 00 00 FE FF 01 00 FF FF
8BD0- 41 01 FE FE 01 01 FF FF
8BD8- 01 01 FE FF 01 01 FE FE
8BE0- 01 01 FE FE 01 00 FF FE
8BE8- 00 00 FE FE 01 01 FE FE
8BF0- 61 01 FE FE 01 00 FE FE
8BF8- 00 00 FE FE 01 00 FE FF
8C00- 81 01 FE FE 01 00 FF FF
8C08- 01 01 A9 4C 8D F5 03 A9
8C10- 1C 8D F6 03 A9 8C 8D F7
8C18- 03 4C B3 FE C9 50 D0 03
8C20- 4C 3E 8C C9 4C D0 03 4C
8C28- 4B 8C C9 43 D0 03 4C 27
8C30- 8D C9 41 D0 03 4C 00 8C
8C38- 4C 00 8E 4C C9 DE 20 B1
8C40- 00 D0 F8 20 58 FC A9 1C
8C48- 85 21 60 A5 B8 85 FC A5
8C50- B9 85 FD 20 B1 00 90 FB
8C58- D0 E1 A5 FC 85 B8 A5 FD
8C60- 85 B9 E6 B8 D0 02 E6 B9
8C68- 20 67 DD B0 CE 20 52 E7

8C70- A9 F9 C5 51 B0 03 4C 99
8C78- E1 20 1A D6 B0 03 4C 0C
8C80- 8D 20 FB DA 20 FB DA A9
8C88- 20 20 0F 8D A9 20 20 0F
8C90- 8D A5 51 A6 50 20 24 ED
8C98- A5 24 85 FE A9 00 85 FF
8CA0- A9 20 20 0F 8D A0 03 20
8CA8- 21 8D B1 9B D0 03 4C E9
8CB0- 8C 10 30 38 E9 7F AA 84
8CB8- 85 A0 D0 84 9D A0 CF 84
8CC0- 9E A0 FF CA F0 07 20 19
8CC8- 8D 10 FB 30 F6 A9 20 20
8CD0- 0F 8D 20 19 8D 30 05 20
8CD8- 0F 8D D0 F6 20 0F 8D A4
8CE0- 85 A9 20 20 0F 8D 4C A7
8CE8- 8C A5 FF A4 FE 20 F2 E2
8CF0- 20 63 EB A4 21 20 01 E3
8CF8- 20 69 EA 20 FB E6 8A 19
8D00- FF 18 65 25 30 06 C9 18
8D08- 10 02 85 25 4C B3 FE 20
8D10- 5C DB E6 FE D0 02 E6 FF
8D18- 60 C8 D0 02 E6 9E B1 9D
8D20- 60 C8 D0 02 E6 9C 60 A0
8D28- 00 A5 67 85 FE 18 69 03
8D30- 85 FA 85 FC A5 68 85 FF
8D38- 69 00 85 FB 85 FD 20 C4
8D40- 8D D0 36 20 BB 8D 20 DC
8D48- 8D A2 02 85 F9 20 C4 8D
8D50- 20 BB 8D CA D0 F5 05 F9
8D58- F0 0E A2 02 20 C4 8D 20
8D60- BB 8D CA D0 F7 4C 3E 8D
8D68- 20 BB 8D 20 BB 8D A5 FC
8D70- 85 AF A5 FD 85 B0 4C B3
8D78- FE C9 B2 F0 06 20 BB 8D
8D80- 4C 3E 8D A9 03 18 65 FE
8D88- 08 C5 FC D0 13 A9 00 28
8D90- 65 FF C5 FD D0 0A A9 04
8D98- 85 F9 20 AA 8D 4C 49 8D
8DA0- A9 01 85 F9 20 AA 8D 4C
8DA8- 43 8D A5 FC 38 E5 F9 B0
8DB0- 02 C6 FD 85 FC 20 C4 8D
8DB8- D0 FB 60 E6 FC D0 02 E6
8DC0- FD 91 FC 60 E6 FA D0 02
8DC8- E6 FB A5 FB C5 B0 D0 09
8DD0- A5 FA C5 AF D0 03 4C 68
8DD8- 8D B1 FA 60 A5 FC 18 69
8DE0- 01 91 FE A5 FD 69 00 C8
8DE8- 91 FE 88 85 FF A5 FC 85
8DF0- FE E6 FE 60

8AA8- 18 E5 3A 85 3E 10 01 C8
8AB0- 98 E5 3B D0 6B A4 2F B9
8AB8- 3D 00 91 3A 88 10 F8 20
8AC0- 1A FC 20 1A FC 20 D0 F8
8AC8- 20 53 F9 84 3B 85 3A 4C
8AD0- 2F 8B 20 BE FF A4 34 20
8AD8- A7 FF 84 34 A0 17 88 30
8AE0- 4B D9 CC FF D0 F8 C0 15
8AE8- D0 E8 A5 31 A0 00 C6 34
8AF0- 20 00 FE 4C 2F 8B A5 3D
8AF8- 20 8E F8 AA BD 00 FA C5
8B00- 42 D0 13 BD C0 F9 C5 43
8B08- D0 0C A5 44 A4 2E C0 9D
8B10- F0 88 C5 2E F0 9F C6 3D
8B18- D0 DC E6 44 C6 35 F0 D6
8B20- A4 34 98 AA 20 4A F9 A9
8B28- DE 20 ED FD 20 3A FF A9
8B30- A1 85 33 20 67 FD 20 C7
8B38- FF AD 00 02 C9 A0 F0 13
8B40- C8 C9 A4 F0 92 88 20 A7
8B48- FF C9 93 D0 D5 8A F0 D2
8B50- 20 78 FE A9 03 85 3D 20
8B58- CE 8B 0A E9 BE C9 C2 90
8B60- C1 0A 0A A2 04 0A 26 42
8B68- 26 43 CA 10 F8 C6 3D F0
8B70- F4 10 E4 A2 05 20 CE 8B
8B78- 84 34 DD B4 F9 D0 13 20
8B80- CE 8B DD BA F9 F0 0D BD
8B88- BA F9 F0 07 C9 A4 F0 03
8B90- A4 34 18 88 26 44 E0 03
8B98- D0 0D 20 A7 FF A5 3F F0
8BA0- 01 E8 86 35 A2 03 88 86
8BA8- 3D CA 10 C9 A5 44 0A 0A
8BB0- 05 35 C9 20 B0 06 A6 35
8BB8- F0 02 09 80 85 44 84 34
8BC0- B9 00 02 C9 BB F0 04 C9
8BC8- 8D D0 80 4C F6 8A B9 00
8BD0- 02 C8 C9 A0 F0 F8 60 20
8BD8- 7D F4 A5 F8 10 13 C9 8E
8BE0- D0 F5 24 F9 10 0A A5 FB
8BE8- F0 06 F6 FA D0 02 E6 F9
8BF0- 60 A9 00 85 F9 85 FA 60
8BF8- FF FF FF FF FF FF FF FF
8C00- 4C 2C 8B 84 58 86 57 85
8C08- 56 08 A9 4C 8D F5 03 A9
8C10- 1C 8D F6 03 A9 8C 8D F7
8C18- 03 4C B3 FE C9 50 D0 03
8C20- 4C 3E 8C C9 4C D0 03 4C
8C28- 4B 8C C9 43 D0 03 4C 27
8C30- 8D C9 41 D0 03 4C 00 8C
8C38- 4C 00 8E 4C C9 DE 20 B1
8C40- 00 D0 F8 20 58 FC A9 1C
8C48- 85 21 60 A5 B8 85 FC A5
8C50- B9 85 FD 20 B1 00 90 FB
8C58- D0 E1 A5 FC 85 B8 A5 FD
8C60- 85 B9 E6 B8 D0 02 E6 B9
8C68- 20 67 DD B0 CE 20 52 E7
8C70- A9 F9 C5 51 B0 03 4C 99
8C78- E1 20 1A D6 B0 03 4C 0C
8C80- 8D 20 FB DA 20 FB DA A9
8C88- 20 20 0F 8D A9 20 20 0F
8C90- 8D A5 51 A6 50 20 24 ED
8C98- A5 24 85 FE A9 00 85 FF
8CA0- A9 20 20 0F 8D A0 03 20

Récapitulation UTILITAIRES

Est constitué des fichiers
UTILITAIRES OBJ, MINI, RENUM.
C'est le seul fichier nécessaire
à l'obtention des 4 fonctions.

8A8E- EA A9
8A90- 8E 85 73 A9 8A 85 74 4C
8A98- 0A 8C E9 81 4A D0 14 A4
8AA0- 3F A6 3E D0 01 88 CA 8A

8CA8- 21 8D B1 9B D0 03 4C E9
8CB0- 8C 10 30 38 E9 7F AA 84
8CB8- 85 A0 D0 84 9D A0 CF 84
8CC0- 9E A0 FF CA F0 07 20 19
8CC8- 8D 10 FB 30 F6 A9 20 20
8CD0- 0F 8D 20 19 8D 30 05 20
8CD8- 0F 8D D0 F6 20 0F 8D A4
8CE0- 85 A9 20 20 0F 8D 4C A7
8CE8- 8C A5 FF A4 FE 20 F2 E2
8CF0- 20 63 EB A4 21 20 01 E3
8CF8- 20 69 EA 20 FB E6 8A 49
8D00- FF 18 65 25 30 06 C9 18
8D08- 10 02 85 25 4C B3 FE 20
8D10- 5C DB E6 FE D0 02 E6 FF
8D18- 60 C8 D0 02 E6 9E B1 9D
8D20- 60 C8 D0 02 E6 9C 60 A0
8D28- 00 A5 67 85 FE 18 69 03
8D30- 85 FA 85 FC A5 68 85 FF
8D38- 69 00 85 FB 85 FD 20 C4
8D40- 8D D0 36 20 BB 8D 20 DC
8D48- 8D A2 02 85 F9 20 C4 8D
8D50- 20 BB 8D CA D0 F5 05 F9
8D58- F0 0E A2 02 20 C4 8D 20
8D60- BB 8D CA D0 F7 4C 3E 8D
8D68- 20 BB 8D 20 BB 8D A5 FC
8D70- 85 AF A5 FD 85 B0 4C B3
8D78- FE C9 B2 F0 06 20 BB 8D
8D80- 4C 3E 8D A9 03 18 65 FE
8D88- 08 C5 FC D0 13 A9 00 28
8D90- 65 FF C5 FD D0 0A A9 04
8D98- 85 F9 20 AA 8D 4C A9 8D
8DA0- A9 01 85 F9 20 AA 8D 4C
8DA8- 43 8D A5 FC 38 E5 F9 B0
8DB0- 02 C6 FD 85 FC 20 C4 8D
8DB8- D0 FB 60 E6 FC D0 02 E6
8DC0- FD 91 FC 60 E6 FA D0 02
8DC8- E6 FB A5 FB C5 B0 D0 09
8DD0- A5 FA C5 AF D0 03 4C 68
8DD8- 8D B1 FA 60 A5 FC 18 69
8DE0- 01 91 FE A5 FD 69 00 C8
8DE8- 91 FE 88 85 FF A5 FC 85
8DF0- FE E6 FE 60 8B 98 C8 B9
8DF8- 00 98 C9 20 F0 06 C9 0D
8F00- A4 B8 8C BB 94 A4 B9 8C
8E08- BC 94 20 BA 94 C9 4D D0
8E10- 03 4C 23 93 A0 01 R1 67
8E18- D0 04 88 4C A0 92 A9 FF
8E20- A2 0A A0 00 85 9D 85 9E
8E28- 84 9F 84 A0 86 A1 84 A2
8E30- 86 A3 84 A4 20 BA 94 90
8E38- 29 F0 57 C9 48 D0 03 4C
8E40- AC 92 C9 43 D0 0D 4C A3
8E48- 93 20 BA 94 C9 2C D0 12
8E50- 20 B2 94 A2 03 DD C9 94
8E58- F0 0D CA 10 F8 EB C9 80
8E60- F0 05 A0 CF 4C A0 92 8A
8E68- 0A A8 20 B2 94 F0 06 90
8E70- 0A C9 2C D0 F5 A2 00 86
8E78- A8 F0 05 20 D2 93 A6 A7
8E80- 96 9D A6 A8 96 9E E0 FA
8E88- 90 05 A0 D6 4C A0 92 AA
8E90- D0 B7 A5 A1 05 A2 D0 05
8E98- A0 EC 4C A0 92 A2 03 B5
8EA0- 67 95 A5 CA 10 F9 86 AB

8EA8- A0 02 B1 A5 C8 C5 9F B1
8EB0- A5 C8 E5 A0 B0 47 20 17
8EB8- 94 D0 ED A0 15 4C A0 92
8ECO- A2 00 86 AB A0 02 B1 A5
8EC8- 81 A7 C8 20 26 94 C0 03
8ED0- F0 F4 A5 A3 81 A7 20 26
8ED8- 94 A5 A4 81 A7 20 26 94
8EE0- 20 17 94 F0 2E 18 A5 A3
8EE8- 65 A1 85 A3 A5 A4 65 A2
8EF0- 85 A4 B0 04 C9 FA 90 05
8EF8- A0 27 4C A0 92 A5 9D A0
8F00- 02 D1 A5 C8 A5 9E F1 A5
8F08- B0 B6 A5 AB 10 05 A0 15
8F10- 4C A0 92 A2 00 A9 FF 81
8F18- A7 20 26 94 A9 FF 81 A7
8F20- 20 26 94 A5 67 85 A5 A5
8F28- 68 85 A6 A0 02 84 AB B1
8F30- A5 85 9D C8 B1 A5 85 9E
8F38- 20 79 94 B0 0E A0 00 84
8F40- AB 20 79 94 90 A0 00 3D
8F48- 4C A0 92 20 17 94 D0 DB
8F50- A5 A7 85 A5 A5 A8 85 A6
8F58- A5 67 8D BB 94 A5 68 8D
8F60- BC 94 A5 69 85 A3 A5 6A
8F68- 85 A4 D0 0C C0 EF 90 05
8F70- A0 DE 4C A0 92 20 B2 94
8F78- 20 B2 94 A8 D0 03 4C 19
8F80- 90 20 B2 94 20 B2 94 A0
8F88- 04 20 B2 94 C8 F0 E1 AA
8F90- F0 DA 10 F5 A2 06 DD CD
8F98- 94 F0 05 CA 10 F8 30 E9
8FA0- 20 B2 94 B0 E7 20 D2 93
8FA8- A5 A7 85 9D A5 A8 85 9E
8FB0- 20 3E 94 A2 00 86 A9 86
8FB8- AA 86 AB A2 0F 06 9D 26
8FC0- 9E F8 A5 A9 65 A9 85 A9
8FC8- A5 AA 65 AA 85 AA A5 AB
8FD0- 65 AB 85 AB D8 CA 10 E5
8FD8- E8 86 9D A2 02 A9 01 85
8FE0- 9E B5 A9 4A 4A 4A 29
8FE8- 0F C5 9D F0 08 C6 9D C8
8FF0- D0 03 4C 70 8F B5 A9 C6
8FF8- 9E F0 EC CA 10 DF A5 9D
9000- D0 03 C8 F0 ED 20 BA 94
9008- C9 C9 F0 07 C9 2C F0 03
9010- 4C 8C 8F C8 F0 DC 4C A0
9018- 8F A2 FF 18 B5 74 95 9E
9020- F5 A6 95 A2 E8 F0 F5 B0
9028- 05 A0 52 4C A0 92 E9 02
9030- B0 0F A0 96 20 A4 94 20
9038- 0C FD C9 D9 F0 03 4C 86
9040- 92 A5 A5 85 9F A5 A6 85
9048- A0 A0 00 C6 9E C6 A0 88
9050- B1 9F 91 9D 9E 18 65 9F
9058- 45 67 D0 08 A5 A0 69 00
9060- 45 68 F0 05 98 D0 E8 F0
9068- E2 38 A5 69 65 A1 85 A3
9070- A5 6A 65 A2 85 A4 A5 67
9078- 85 A5 65 A1 8D BB 94 A5
9080- 68 85 A6 65 A2 8D BC 94
9088- 90 14 18 98 A0 00 65 A5
9090- 91 A5 C8 AA A5 A6 69 00
9098- 91 A5 86 A5 A5 A6 20 R2
90A0- 94 20 B2 94 A8 D0 03 4C

90A8- 54 91 20 B2 94 85 9D 20
90B0- B2 94 85 9E 20 3E 94 A0
90B8- 02 A5 9D 91 A5 C8 A5 9E
90C0- 91 A5 C8 20 B2 94 91 A5
90C8- C8 AA F0 BE 10 F5 A2 06
90D0- DD CD 94 F0 05 CA 10 F8
90D8- 30 E9 20 B2 94 B0 E7 20
90E0- D2 93 A5 A7 85 9D A5 A8
90E8- 85 9E 20 3E 94 A2 00 86
90F0- A9 86 AA 86 AB A2 0F 06
90F8- 9D 26 9E F8 A5 A9 65 A9
9100- 85 A9 A5 AA 65 AA 85 AA
9108- A5 AB 65 AB 85 AB D8 CA
9110- 10 E5 E8 86 9D A2 02 A9
9118- 01 85 9E B5 A9 4A 4A 4A
9120- 4A 29 0F C5 9D F0 07 C6
9128- 9D 09 30 91 A5 C8 B5 A9
9130- C6 9E F0 ED CA 10 E0 A5
9138- 9D D0 05 A9 30 91 A5 C8
9140- 20 BA 94 C9 C9 F0 07 C9
9148- 2C F0 03 4C C6 90 91 A5
9150- C8 4C DA 90 18 A5 A5 69
9158- 02 85 69 85 AF A5 A6 69
9160- 00 85 6A 85 B0 A9 00 A0
9168- 07 91 A5 88 10 FB 38 A5
9170- 73 E9 06 85 9D A8 A5 74
9178- E9 00 85 9E C4 69 E5 6A
9180- E9 01 B0 05 A0 52 4C A0
9188- 92 A0 05 A9 00 91 9D 88
9190- 10 FB A5 67 85 9F 85 A5
9198- A5 68 85 A0 85 A6 A0 00
91A0- 84 A1 84 A2 C8 20 23 94
91A8- D0 03 4C 86 92 A0 02 A5
91B0- A1 D1 A5 C8 A5 A2 F1 A5
91B8- B0 11 B1 A5 85 A2 88 B1
91C0- A5 85 A1 A5 A5 85 9F A5
91C8- A6 85 A0 20 17 94 D0 DD
91D0- A0 00 38 B1 9F 85 A5 E5
91D8- 9F 85 A3 C8 B1 9F 85 A6
91E0- E5 A0 85 A4 A5 9D 85 A7
91E8- E5 A3 85 9D A5 9E 85 A8
91F0- E5 A4 85 9E A4 A3 88 B1
91F8- 9F 91 9D 88 D0 F9 B1 9F
9200- 91 9D A5 A5 85 A7 A5 A6
9208- 85 A8 20 17 94 A0 00 B1
9210- A7 91 9F C8 AA D0 F8 C0
9218- 05 90 F4 A5 A5 05 A6 F0
9220- 16 18 98 A0 00 65 9F 91
9228- 9F AA C8 A9 00 65 A0 91
9230- 9F 86 9F 85 A0 D0 CB A0
9238- 01 B1 67 F0 03 4C 92 91
9240- A5 67 85 A5 A5 68 85 A6
9248- 38 A5 9D E9 01 8D BB 94
9250- A5 9E E9 00 8D BC 94 A0
9258- 00 20 B2 94 91 A5 C8 AA
9260- D0 F7 C0 05 90 F3 84 AB
9268- A0 01 20 23 94 F0 17 18
9270- A0 00 A5 AB 65 A5 91 A5
9278- AA C8 A9 00 65 A6 91 A5
9280- 86 A5 85 A6 D0 D1 A9 EF
9288- 85 C1 A2 0E A9 00 95 9D
9290- 9D 00 02 CA 10 F8 85 B8
9298- A2 02 86 B9 4C 6C D6 60
92A0- 20 2D FF 20 48 F9 20 A4

92A8- 94 4C 86 92 AD D3 94 0D
 92B0- D4 94 F0 05 A0 85 4C A0
 92B8- 92 A5 73 C5 AF A5 74 E5
 92C0- B0 E9 04 B0 05 A0 52 4C
 92C8- A0 92 A5 AF E5 67 8D D3
 92D0- 94 A5 B0 E5 68 8D D4 94
 92D8- A5 73 8D D5 94 A5 74 8D
 92E0- D6 94 A0 00 A5 73 D0 02
 92E8- C6 74 C6 73 A5 AF D0 02
 92F0- C6 B0 C6 AF B1 AF 91 73
 92F8- A5 67 C5 AF A5 68 E5 B0
 9300- 90 E2 A5 67 69 02 85 AF
 9308- 85 69 A5 68 69 00 85 B0
 9310- 85 6A A0 01 A9 00 91 67
 9318- 88 10 FB A0 60 20 A4 94
 9320- 4C 86 92 AD D3 94 0D D4
 9328- 94 F0 F5 38 A4 73 A5 74
 9330- E9 01 C4 69 E5 6A B0 05
 9338- A0 52 4C A0 92 A5 67 A6
 9340- 68 85 A1 86 A2 A0 01 B1
 9348- A1 F0 07 AA 88 B1 A1 4C
 9350- 41 93 A5 A1 85 42 A5 A2
 9358- 05 43 AD D5 94 85 3E AD
 9360- D6 94 85 3F A5 73 85 3C
 9368- A5 74 85 3D A0 00 20 2C
 9370- FE AD D5 94 85 73 AD D6
 9378- 94 85 74 38 AD D3 94 E9
 9380- 02 85 A1 AD D4 94 E9 00
 9388- 85 A2 18 A5 AF 65 A1 85
 9390- AF 85 69 A5 B0 65 A2 85
 9398- B0 85 6A A9 00 8D D3 94
 93A0- 8D D4 94 A5 67 85 A1 A5
 93A8- 68 85 A2 18 A0 01 B1 A1
 93B0- F0 1D A0 04 C8 B1 A1 D0
 93B8- FB C8 98 65 A1 AA A0 00
 93C0- 91 A1 A5 A2 69 00 C8 91
 93C8- A1 86 A1 85 A2 90 DD 4C
 93D0- 65 91 A2 00 86 A7 86 A8
 93D8- AA 06 A7 26 A8 B0 31 A5
 93E0- A8 85 AB A5 A7 0A 26 AB
 93E8- B0 26 0A 26 AB B0 21 65
 93F0- A7 85 A7 A5 AB 65 A8 85
 93F8- A8 B0 15 8A 29 0F 65 A7
 9400- 85 A7 A5 A8 69 00 85 A8

9408- B0 06 20 B2 94 90 C9 60
 9410- 68 68 A0 D6 4C A0 92 A0
 9418- 00 B1 A5 C8 AA A1 A5 86
 9420- A5 85 A6 B1 A5 60 E6 A7
 9428- D0 02 E6 A8 A5 A7 C5 73
 9430- A5 A8 E5 74 90 07 68 68
 9438- A0 52 4C A0 92 60 84 AB
 9440- A5 A3 85 A7 A5 A4 85 A8
 9448- A0 00 B1 A7 45 9D D0 07
 9450- C8 B1 A7 45 9E F0 15 18
 9458- A5 A7 69 04 85 A7 90 02
 9460- E6 A8 A0 01 B1 A7 C9 FF
 9468- D0 DE F0 0A C8 B1 A7 85
 9470- 9D C8 B1 A7 85 9E A4 AB
 9478- 60 A5 69 85 A3 A5 6A 85
 9480- A4 A4 AB 18 B1 A3 45 9D
 9488- D0 07 C8 B1 A3 45 9E F0
 9490- 12 A5 A3 69 04 85 A3 90
 9498- 02 E6 A4 A0 01 B1 A3 C9
 94A0- FF D0 DE 60 B9 00 95 48
 94A8- C8 09 80 20 ED FD 68 10
 94B0- F3 60 EE BB 94 D0 03 EE
 94B8- BC 94 AD 8C 8A C9 3A B0
 94C0- 06 38 E9 30 38 E9 D0 60
 94C8- 00 45 53 49 46 85 AC AB
 94D0- B0 BC C4 00 00 00 00 18
 94D8- A5 A3 65 A1 85 A3 A5 A4
 94E0- 65 A2 85 A4 B0 04 C9 FA
 94E8- 90 05 A0 27 4C A0 92 A5
 94F0- 9D A0 02 D1 A5 C8 A5 9E
 94F8- F1 A5 B0 B6 A5 AB 10 05
 9500- 4E 4F 20 50 52 4F 47 52
 9508- 41 4D 20 49 4E 20 4D 45
 9510- 4D 4F 52 59 8D 4E 4F 20
 9518- 4C 49 4E 45 53 20 49 4E
 9520- 20 52 41 4E 47 45 8D 49
 9528- 4E 43 52 45 4D 45 4E 54
 9530- 20 54 4F 4F 20 4C 41 52
 9538- 47 45 8D 44 55 50 4C 49
 9540- 43 41 54 45 20 4C 49 4F
 9548- 45 20 4E 55 4D 42 45 52
 9550- 53 8D 4F 55 54 20 4F 46
 9558- 20 4D 45 4D 4F 52 59 8D
 9560- 50 52 4F 47 52 41 4D 20

9568- 4F 4E 20 48 4F 4C 44 2C
 9570- 20 55 53 45 20 22 26 4D
 9578- 22 20 54 4F 20 52 45 43
 9580- 4F 56 45 52 8D 48 4F 4C
 9588- 44 20 46 49 4C 45 20 49
 9590- 4E 20 55 53 45 8D 4C 49
 9598- 4D 49 54 45 44 20 4D 45
 95A0- 4D 4F 52 59 2C 20 4D 41
 95A8- 59 20 44 45 53 54 52 4F
 95B0- 59 20 50 52 4F 47 52 41
 95B8- 4D 20 20 20 20 20 43 4F
 95C0- 4E 54 49 4E 55 45 20 28
 95C8- 59 2F 4E 29 3F 07 8D 53
 95D0- 59 4E 54 41 58 8D 3E 20
 95D8- 36 33 39 39 39 8D 4C 49
 95E0- 4E 45 20 54 4F 4F 20 4C
 95E8- 4F 4E 47 9B 49 4E 43 52
 95F0- 45 4D 45 4E 54 20 3D 20
 95F8- 30 8D C9 C9 F0 07 C9 00
 9600- 46

Récapitulation MINI

Sert à constituer le fichier UTILITAIRES. Il est obtenu à partir du fichier INTBASIC de la disquette Master ; se reporter au texte.

Récapitulation RENUM

Sert à constituer le fichier UTILITAIRES. Il est obtenu à partir du fichier RENUMBER de la disquette Master ; se reporter au texte.

Max : le moniteur étendu

Jacques Supernant

Apple][+, //e, //c

Ce moniteur autorise un contrôle de l'exécution des routines en langage machine.

Un mode Trace et Pas à Pas très évolués et sélectifs sont complétés par un accès direct aux registres du 6502 (ou 65C02).

La gestion des fenêtres d'écran simplifie le mode trace. Une routine permet la recherche de suites d'octets.

Un mini-assembleur très souple fait partie de Max. Une ligne de commande peut devenir une boucle avec l'ordre JUMP

Les fichiers source sont sur la disquette.

Disquette et documentation : 150,00 F TTC franco. Bon de commande p 74.

Utilisation de la carte langage

Hervé Roy-Contancin

Afin de rassurer pleinement les lecteurs de Pom's redoutant l'abandon de la ligne 6502, voici un programme qui va permettre aux possesseurs d'Apple (II et II+ avec carte

langage, //e et //c) d'utiliser cette carte en DOS 3.3 sous Applesoft. Les 16 Ko de la carte langage, normalement inaccessibles en Applesoft, peuvent enfin être utilisés pour y stocker une ou plusieurs tables alphanumériques.

Il est important de noter que toutes les chaînes de caractères doivent avoir la même longueur, cette longueur étant fixée au départ (voir ci-dessous).

Pratiquement...

- En premier lieu, il faut charger le programme par :

BLOAD LANG1.OBJ0,A\$9400

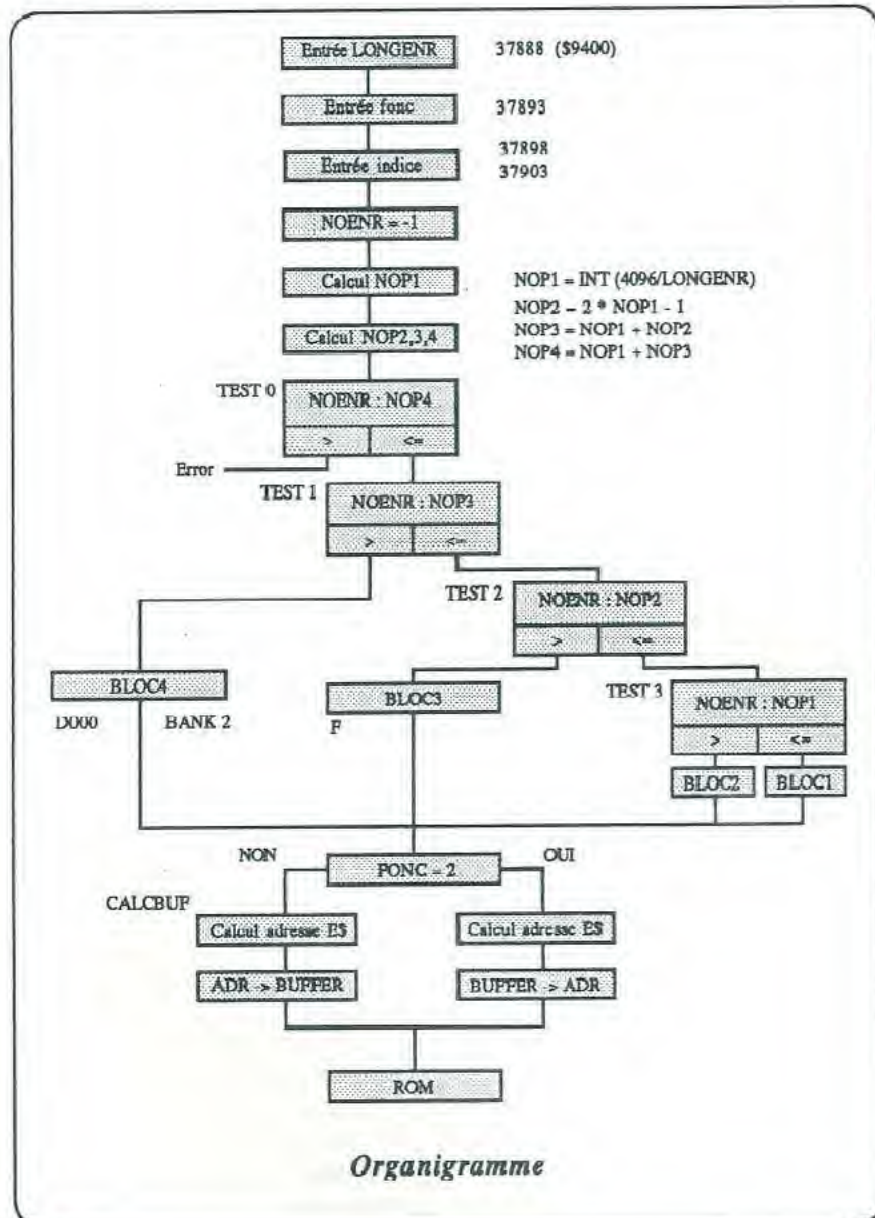
- puis placer une HIMEM: 37888 afin de ne pas écraser le programme LANG1.OBJ0 par les variables du programme Applesoft.

- il faut ensuite définir une variable E\$ en l'initialisant à la longueur des chaînes de la table. Si la longueur est 8, faire par exemple E\$ = "BONJOUR!". Cette variable E\$ est indispensable ; elle doit être la première variable rencontrée dans le programme Applesoft ; enfin, on ne doit jamais l'écourter mais il est possible de rajouter des blancs pour conserver la même longueur : E\$ = "HELLO ".

- On peut écrire ou lire une chaîne sur la carte langage en faisant :

CALL 37888,LNG,FONC,IH,IB
où LNG est la longueur des chaînes (c'est-à-dire la longueur initiale de E\$), FONC vaut 1 en lecture, 2 en écriture, IH est l'indice haut et IB l'indice bas du numéro de la chaîne (c'est-à-dire son numéro est $IH * 256 + IB$). Ce numéro doit être compris entre 1 et $4 * (4096 / LNG)$ car la carte langage peut stocker 4 fois 4 Ko et 4Ko = 4096 octets.

L'indice IH peut aussi être



considéré comme un numéro de table dans une gestion de 1 à 4 tables de 256 chaînes.

Par exemple, la lecture de la 39ème chaîne donne, si la longueur est 32 :

CALL 37888,32,1,0,39 : PRINT E\$

Ceci est équivalent à PRINT E\$(39) pour un tableau de chaînes en Applesoft.

Comme le montre le programme EXEMPLE-LANG1, il est possible d'abrégier l'écriture de la syntaxe si certains paramètres restent inchangés, mais il faut alors changer l'adresse du CALL.

CALL 37888,LNG,FONC,IH,IB
(est obligatoire pour le premier appel)

CALL 37894,FONC,IH,IB
(pour les autres appels)

CALL 37900,IH,IB
(si FONC ne change pas)

CALL 37906,IB

(si IH ne change pas)

CALL 37912

(si rien a changé)

Des contrôles sont effectués et les messages d'erreur sont les suivants :

- si FONC <> 1 et FONC <> 2 : cela correspond à un ILLEGAL QUANTITY ERROR
- si $IH * 256 + IB > INT(4096 / LNG) * 4$ on obtient un OUT OF MEMORY ERROR
- si c'est une erreur sur E\$: STRING TOO LONG ERROR (en fait c'est l'inverse, mais le message STRING TOO SHORT n'existe pas !).

Techniquement...

L'adresse de début choisie (\$9400, soit 37888 en décimal) peut être modifiée, à condition de modifier les quelques JSR du programme. Il faut bien faire

attention car 3 octets sont réservés, à la fin de LANG1.OBJ0, pour y stocker des variables, afin de ne pas encombrer la page zéro. Le programme gère la mémoire en 4 blocs de 4 Ko, les chaînes y sont enregistrées bout à bout.

Après la saisie des paramètres par la routine GETBYTC, le programme calcule le nombre d'enregistrements par blocs, soit $4096 / LNG$ et on déduit le dernier numéro de chaque bloc (NOP1, NOP2, NOP3 et NOP4). Il calcule ensuite ADR, adresse concernée sur la carte langage, en fonction du bloc auquel appartient l'enregistrement demandé (\$D000 - \$DFFF/BANK1 pour le BLOC1, \$D000 - \$DFFF/BANK2 pour le BLOC4, \$E000 - \$EFFF pour le BLOC2 et \$F000 - \$FFFF pour le BLOC3). Enfin, BUFFER est l'adresse de la variable E\$.

A noter que la fonction lecture

Programme EXEMPLE-LANG1

```

10 REM EXEMPLE
20 REM *****
30 REM * PROGRAMME DE DEMONSTRATION
40 REM * UTILISATION DE LANG1.OBJ0
50 REM * H.ROY-CONTANCIN
60 REM *****
70 HIMEM: 37888: REM INDISPENSABLE
75 E$ = "JE SUIS LE NUMERO 39 HA HA HA HA":D$ =
  CHR$(4)
76 REM E$ DOIT ETRE LA PREMIERE VARIABLE DU
  PGM
80 PRINT D$"BLOAD LANG1.OBJ0,A$9400"
90 BL$ = "": REM 32
  BLANKS
100 LNG = 32:ECR = 2:LEC = 1
110 CALL 37888,LNG,ECR,0,39: REM ECRITURE DU 39
120 E$ = BL$: REM ON EFFACE
130 CALL 37888,LNG,LEC,0,39: PRINT E$: REM
  RELECTURE
140 REM ET MAINTENANT 259 > 255
150 I = 259:E$ = "ET MOI ET MOI LE 259 ME VOICI OK"
155 CALL 37888,LNG,ECR,INT(I/256),I-INT(I/256)*
  256
160 E$ = BL$
170 CALL 37888,LNG,LEC,INT(I/256),I-INT(I/256)*
  256: PRINT E$
180 E$ = BL$
190 REM POUR LES PARESSEUX
200 CALL (37888+6),LEC,INT(I/256),I-INT(I/256)*
  256: PRINT E$
210 REM ENCORE PLUS COURT

```

```

220 E$ = BL$: CALL (37888+12),0,39: PRINT E$
230 REM ET POUR FINIR
235 E$ = BL$
240 CALL (37888+18),39: PRINT E$
250 END

```

Programme LANG1 (Assembleur Toolkit)

SOURCE FILE: LANG1

```

0000:      1 *****
0000:      2 * LANG1
0000:      3 * CARTE LANGAGE BASIC *
0000:      4 * 03/02/85
0000:      5 * H.ROY-CONTANCIN
0000:      6 *****
000B:      7 NOP1 EQU $0B
0018:      8 NOP2 EQU $18
001A:      9 NOP3 EQU $1A
00F9:     10 NOP4 EQU $F9
001C:     11 BUFFER EQU $1C
001E:     12 ADR EQU $1E
C082:     13 ROM EQU $C082
C083:     14 RAM EQU $C083
0050:     15 LINUM EQU $50
E752:     16 GETADR EQU $E752
E6F5:     17 GETBYTC EQU $E6F5
E982:     18 FMULTT EQU $E982
E2F2:     19 GIVAYF EQU $E2F2
EB63:     20 MOVEAF EQU $EB63
EB53:     21 MOVEFA EQU $EB53

```


effectue le filtrage des caractères de code ASCII non compris entre \$20 et \$60 pour éviter tout problème en cas de lecture d'une zone de la mémoire non initialisée.

Si E\$ est initialisée par E\$ = "A...A" (32 A), puis si B\$ = "<50 blancs>", le fait d'écrire E\$ = B\$ attribue à E\$ l'adresse de la chaîne B\$: ainsi, le résultat de la recherche se trouve alors dans B\$ et E\$ a une longueur 50 (E\$="A...A ", 32 A et 18 blancs). Pour éviter cela, il suffit de remplacer E\$ = B\$ par E\$ = B\$ + "".

Remarquons, pour conclure, que ce programme permet non seulement de gagner de la place mémoire, mais également de transmettre des tables entre plusieurs programmes puisque la carte langage n'est pas touchée par RUN ou LOAD. Cette seule raison peut justifier son emploi en zone de sauvegarde de tables.

NB : J'ai dû utiliser la routine ABS après les routines FDI VT (\$EA69) et FMULTT (\$E982) car dans certains cas $4096 / 32 = 128$!

Carte langage 16 Ko		
Adresses	Bank 1	Bank 2
D000 D1FF	BLOC 1	BLOC 4
E000 EFFF	BLOC 2	
F000 FFFF	BLOC 3	

N.D.L.R.

• Nous avons fait des essais de vitesse pour la lecture et l'écriture de 450 chaînes de caractères :

- Avec le programme LANG1, il faut 42 secondes;
- Avec le Disque virtuel 16K RWLC (Pom's 12) et un fichier à accès direct, 111

secondes sont nécessaires ;

- Et enfin, pour Tableau de chaînes Applesoft : il est préférable d'utiliser FRE(64) après le test (Pom's 2) : cela prend 39 secondes.

• C'est une méthode originale pour utiliser la carte langage en Applesoft pour y stocker des chaînes de longueur fixe. A titre de comparaison, on connaissait déjà :

- le DOS sur la carte langage, ce qui remonte la HIMEM et laisse plus de place pour les programmes.
- le Disque virtuel 16 Ko (en fait, il n'y a que 15K 1/4 pour les données, car il faut aussi stocker le catalogue.
- Haifa 64K : on double les instructions de l'Applesoft.



```

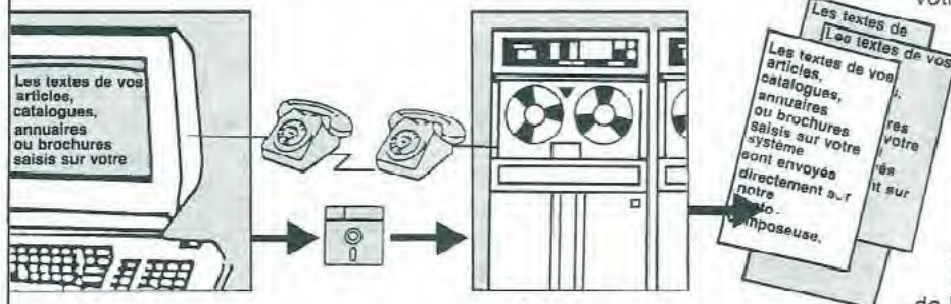
EA69:      22 FDI VT   EQU  $EA69      9437:18      52 CNOP3   CLC
EBAF:      23 ABS    EQU  $EBAF      9438:A5 0B      53          LDA  NOP1
EC23:      24 INT    EQU  $EC23      943A:65 18      54          ADC  NOP2
D419:      25 ERROR  EQU  $D419      943C:85 1A      55          STA  NOP3
0069:      26 LOMEM  EQU  $69        943E:A5 0C      56          LDA  NOP1+1
0000:      27 *****
----- NEXT OBJECT FILE NAME IS LANG1.OBJO
9400:      28          ORG  $9400      9444:A5 0B      59 CNOP4   LDA  NOP1
9400:20 F5 E6      29 DEB    JSR  GETBYTC  9446:18        60          CLC
9403:8E D4 95      30          STX  LONGENR  9447:65 1A      61          ADC  NOP3
9406:20 F5 E6      31 ENTREE JSR  GETBYTC  9449:85 F9      62          STA  NOP4
9409:8E D3 95      32          STX  FONC    944B:A5 0C      63          LDA  NOP1+1
940C:20 F5 E6      33          JSR  GETBYTC  944D:65 1B      64          ADC  NOP3+1
940F:8E D5 95      34          STX  NOENR   944F:85 FA      65          STA  NOP4+1
9412:20 F5 E6      35          JSR  GETBYTC  9451:A9 D0      66          LDA  #$D0
9415:8E D6 95      36          STX  NOENR+1  9453:85 1F      67          STA  ADR+1
9418:38          37          SEC          9455:A9 00      68          LDA  #$00
9419:AD D6 95      38          LDA  NOENR+1  9457:85 1E      69          STA  ADR
941C:E9 01          39          SBC  #$01    9459:A5 FA      70 TEST0   LDA  NOP4+1
941E:8D D6 95      40          STA  NOENR+1  945B:CD D5 95   71          CMP  NOENR
9421:AD D5 95      41          LDA  NOENR   945E:90 43      72          BCC  ERREUR
9424:E9 00          42          SBC  #$00    9460:D0 09      73          BNE  TEST1
9426:8D D5 95      43          STA  NOENR   9462:A5 F9      74          LDA  NOP4
9429:20 9D 95      44          JSR  CALCNOP1  9464:CD D6 95   75          CMP  NOENR+1
942C:18          45 CNOP2   CLC          9467:90 3A      76          BCC  ERREUR
942D:A5 0B          46          LDA  NOP1    9469:F0 38      77          BEQ  ERREUR
942F:2A          47          ROL  A         946B:A5 1B      78 TEST1   LDA  NOP3+1
9430:85 18          48          STA  NOP2    946D:CD D5 95   79          CMP  NOENR
9432:A5 0C          49          LDA  NOP1+1  9470:90 74      80          BCC  BLOC4
9434:2A          50          ROL  A         9472:D0 09      81          BNE  TEST2
9435:85 19          51          STA  NOP2+1  9474:A5 1A      82          LDA  NOP3

```


9476:CD D6 95	83	CMP	NOENR+1	94ED:AD D5 95	142	LDA	NOENR
9479:90 6B	84	BCC	BLOC4	94F0:E5 1B	143	SBC	NOP3+1
947B:F0 69	85	BEQ	BLOC4	94F2:20 66 95	144	JSR	CALCADR
947D:A5 19	86 TEST2	LDA	NOP2+1	94F5:A2 08	145	LDX	#\$08
947F:CD D5 95	87	CMP	NOENR	94F7:	146 * BANK2	(C083+08=C08D)	
9482:90 4A	88	BCC	BLOC3	94F7:AD D3 95	147 TESTFON	LDA	FONC
9484:D0 09	89	BNE	TEST3	94FA:C9 02	148	CMP	#\$02
9486:A5 18	90	LDA	NOP2	94FC:F0 29	149	BEQ	ECRIT
9488:CD D6 95	91	CMP	NOENR+1	94FE:C9 01	150	CMP	#\$01
948B:90 41	92	BCC	BLOC3	9500:D0 3D	151	BNE	ERREUR2
948D:F0 3F	93	BEQ	BLOC3	9502:20 44 95	152 LEC	JSR	CALCBUF
948F:A5 0C	94 TEST3	LDA	NOP1+1	9505:BD 83 C0	153	LDA	RAM, X
9491:CD D5 95	95	CMP	NOENR	9508:BD 83 C0	154	LDA	RAM, X
9494:90 20	96	BCC	BLOC2	950B:AC D4 95	155	LDY	LONGENR
9496:D0 10	97	BNE	BLOC1	950E:88	156 L1	DEY	
9498:A5 0B	98	LDA	NOP1	950F:B1 1E	157	LDA	(ADR), Y
949A:CD D6 95	99	CMP	NOENR+1	9511:	158 * CARACT	INDESIRABLES?	
949D:90 17	100	BCC	BLOC2	9511:C9 20	159	CMP	#\$20
949F:F0 15	101	BEQ	BLOC2	9513:90 0D	160	BCC	BIDON
94A1:B0 05	102	BCS	BLOC1	9515:C9 60	161	CMP	#\$60
94A3:A2 4D	103 ERREUR	LDX	#\$4D	9517:B0 09	162	BCS	BIDON
94A5:4C 19 D4	104	JMP	ERROR	9519:91 1C	163 E11	STA	(BUFFER), Y
94A8:AD D5 95	105 BLOC1	LDA	NOENR	951B:98	164	TYA	
94AB:AC D6 95	106	LDY	NOENR+1	951C:D0 F0	165	BNE	L1
94AE:20 66 95	107	JSR	CALCADR	951E:AD 82 C0	166	LDA	ROM
94B1:A2 00	108	LDX	#\$00	9521:60	167 FIN	RTS	
94B3:	109 * BANK1	(C083+0)		9522:A9 20	168 BIDON	LDA	#\$20
94B3:18	110	CLC		9524:18	169	CLC	
94B4:90 41	111	BCC	TESTFON	9525:90 F2	170	BCC	E11
94B6:A9 E0	112 BLOC2	LDA	#\$E0	9527:20 44 95	171 ECRIT	JSR	CALCBUF
94B8:85 1F	113	STA	ADR+1	952A:BD 83 C0	172	LDA	RAM, X
94BA:AD D6 95	114	LDA	NOENR+1	952D:BD 83 C0	173	LDA	RAM, X
94BD:38	115	SEC		9530:AC D4 95	174	LDY	LONGENR
94BE:E5 0B	116	SBC	NOP1	9533:88	175 E1	DEY	
94C0:A8	117	TAY		9534:B1 1C	176	LDA	(BUFFER), Y
94C1:AD D5 95	118	LDA	NOENR	9536:91 1E	177	STA	(ADR), Y
94C4:E5 0C	119	SBC	NOP1+1	9538:98	178	TYA	
94C6:20 66 95	120	JSR	CALCADR	9539:D0 F8	179	BNE	E1
94C9:A2 00	121	LDX	#\$00	953B:AD 82 C0	180	LDA	ROM
94CB:	122 * BANK1	(C083+0)		953E:60	181	RTS	
94CB:18	123	CLC		953F:A2 35	182 ERREUR2	LDX	#\$35
94CC:90 29	124	BCC	TESTFON	9541:20 19 D4	183	JSR	ERROR
94CE:A9 F0	125 BLOC3	LDA	#\$F0	9544:	184 *****		
94D0:85 1F	126	STA	ADR+1	9544:A0 00	185 CALCBUF	LDY	#\$00
94D2:AD D6 95	127	LDA	NOENR+1	9546:B1 69	186	LDA	(LOMEM), Y
94D5:38	128	SEC		9548:C9 45	187	CMP	#\$45
94D6:E5 18	129	SBC	NOP2	954A:D0 4C	188	BNE	ERREUR1
94D8:A8	130	TAY		954C:C8	189	INY	
94D9:AD D5 95	131	LDA	NOENR	954D:B1 69	190	LDA	(LOMEM), Y
94DC:E5 19	132	SBC	NOP2+1	954F:C9 80	191	CMP	#\$80
94DE:20 66 95	133	JSR	CALCADR	9551:D0 45	192	BNE	ERREUR1
94E1:A2 00	134	LDX	#\$00	9553:C8	193	INY	
94E3:	135 * ENCORE	BANK1		9554:B1 69	194	LDA	(LOMEM), Y
94E3:18	136	CLC		9556:CD D4 95	195	CMP	LONGENR
94E4:90 11	137	BCC	TESTFON	9559:90 3D	196	BCC	ERREUR1
94E6:AD D6 95	138 BLOC4	LDA	NOENR+1	955B:C8	197	INY	
94E9:38	139	SEC		955C:B1 69	198	LDA	(LOMEM), Y
94EA:E5 1A	140	SBC	NOP3	955E:85 1C	199	STA	BUFFER
94EC:A8	141	TAY		9560:C8	200	INY	

Vos textes en direct de votre ordinateur à nos photocomposeuses

Gain de temps et économie



Les textes de vos articles, catalogues, annuaires ou brochures saisis sur votre micro-ordinateur sont envoyés directement sur notre photocomposeuse

Nous vous évitons ainsi, le coût et le temps de la saisie supplémentaire que nécessite le traitement traditionnel de la photocomposition avant l'impression des documents.

Si vous le désirez nous pouvons également nous charger de l'impression et du brochage.

Vous

Nous

TELECOMPO (1) 328.18.63

PHOTOCOMPOSITION - BUREAUTIQUE - TRANSMISSION DE DONNÉES - GESTION DE FICHIERS - MATÉRIELS DE TRAITEMENT DE TEXTES
13 et 15, avenue du Petit-Parc - 94300 VINCENNES

Une référence : la revue
pom's

9561:B1 69	201	LDA (LOMEM),Y	959D:A9 10	231	CALCNOPI LDA #\$10
9563:85 1D	202	STA BUFFER+1	959F:A0 00	232	LDY #\$00
9565:60	203	RTS	95A1:20 F2 E2	233	JSR GIVAYF
9566:20 F2 E2	204	CALCADR JSR GIVAYF	95A4:20 63 EB	234	JSR MOVEAF
9569:20 63 EB	205	JSR MOVEAF	95A7:A9 00	235	LDA #\$00
956C:A9 00	206	LDA #\$00	95A9:AC D4 95	236	LDY LONGENR
956E:AC D4 95	207	LDY LONGENR	95AC:20 F2 E2	237	JSR GIVAYF
9571:20 F2 E2	208	JSR GIVAYF	95AF:20 69 EA	238	JSR FDIPT
9574:20 82 E9	209	JSR FMULTT	95B2:20 23 EC	239	JSR INT
9577:20 23 EC	210	JSR INT	95B5:20 AF EB	240	JSR ABS
957A:20 AF EB	211	JSR ABS	95B8:A5 50	241	LDA LINUM
957D:A5 50	212	LDA LINUM	95BA:48	242	PHA
957F:48	213	PHA	95BB:A5 51	243	LDA LINUM+1
9580:A5 51	214	LDA LINUM+1	95BD:48	244	PHA
9582:48	215	PHA	95BE:20 52 E7	245	JSR GETADR
9583:20 52 E7	216	JSR GETADR	95C1:A5 50	246	LDA LINUM
9586:A5 50	217	LDA LINUM	95C3:85 0B	247	STA NOP1
9588:85 1E	218	STA ADR	95C5:A5 51	248	LDA LINUM+1
958A:A5 51	219	LDA LINUM+1	95C7:85 0C	249	STA NOP1+1
958C:18	220	CLC	95C9:68	250	PLA
958D:65 1F	221	ADC ADR+1	95CA:85 51	251	STA LINUM+1
958F:85 1F	222	STA ADR+1	95CC:68	252	PLA
9591:68	223	PLA	95CD:85 50	253	STA LINUM
9592:85 51	224	STA LINUM+1	95CF:60	254	RTS
9594:68	225	PLA	95D0:4C 19 D4	255	JMP ERROR
9595:85 50	226	STA LINUM	95D3:	256	**** ZONES A RESERVER
9597:60	227	RTS	95D3:	257	FONC EQU *
9598:A2 B0	228	ERREUR1 LDX #\$B0	95D4:	258	LONGENR EQU ++1
959A:4C 19 D4	229	JMP ERROR	95D5:	259	NOENR EQU ++2
959D:	230	*****			*** SUCCESSFUL ASSEMBLY: NO ERRORS

DOS 3.3 ou ProDOS

à la carte

ou comment conserver

Nous avons déjà vu comment faire coexister sur une même disquette des fichiers DOS 3.3 et des fichiers ProDOS. Ceci n'était utile que pour la circulation simultanée des deux types de fichiers, sans que l'on puisse faire d'échanges entre les deux systèmes. Aujourd'hui, nous rendons cela possible mais à une condition : posséder une carte 80 colonnes étendue. (En fait, il est possible de se débrouiller avec ses seuls 64Ko, mais au prix d'une multiplication des accès disques et de la neutralisation d'une bonne partie de mémoire : 21Ko pour les deux DOS).

Utilitaires DOS... ...sous ProDOS

Vous pourrez alors passer quasi instantanément d'un DOS à l'autre tout en gardant en mémoire le programme Basic. Vous pourrez également le faire en cours d'exécution d'un programme (si, si !) et, enfin, vous garderez le bénéfice des éventuels utilitaires que le DOS 3.3 avait mis au chaud sous ses buffers.

Exemple d'utilisation : vous pourrez dorénavant convertir de manière non aveugle. Vous chargez un programme DOS, vous le listez, vous faites les éventuelles modifications nécessitées par l'adaptation à ProDOS et vous le sauvegardez sous ProDOS. Ou encore, vous chargez un programme ProDOS, vous passez sous DOS pour bénéficier des services de RENUMBER, HOLD et MERGE de l'Applesoft Toolkit et vous revenez sauvegarder le résultat sous ProDOS.

L'art et la manière

Il s'agit d'utiliser la mémoire auxiliaire 64 Ko pour stocker les images des deux systèmes d'exploitation. Pour réaliser la permutation, on commencera par sauvegarder l'état de l'OS en fonction, et on appellera son collègue. Le tout à l'aide de la routine AUXMOVE (\$C311).

Quelques petites précautions :

- brancher et débrancher les vecteurs d'interception des entrées sorties, I/O Hooks, des OS (CSSWON et CSSWOFF de DOS 3.3) ;
- rétablir le mode TRACE correct : ProDOS laisse toujours Applesoft en mode TRACE et recopie en page globale BI (\$BE41) la vraie valeur du flag TRACE ;
- de la même manière, établir le lien entre HIMEM DOS 3.3 et HIMEM ProDOS : si pour DOS, la région située au-dessus de HIMEM est bien définitivement protégée (à moins de modifier MAXFILES, ce qui n'est pas souhaitable), il n'en est pas de même pour ProDOS qui maintient un tampon de 1 Ko au dessus de HIMEM à des fins de ramasse-miettes, et surtout qui descend tout ce monde de 1 Ko chaque fois qu'il ouvre un fichier.

Pour résoudre ce dernier problème, il faut recourir à l'allocation de buffer autorisée par ProDOS. La routine correspondante prend en entrée le nombre de pages à allouer et retourne l'adresse de la première page attribuée au tampon. Ce nombre est en fait la valeur du pointeur sur le sommet des tampons de fichiers (TOPBUFF).

Seul l'espace situé au dessus de ce sommet est effectivement protégé de modifications intempestives (voir figure).

Enfin, dernier problème, on a essayé d'utiliser au maximum les adresses situées dans les "pages globales" (\$BE et \$BF), seules celles-ci sont assurées d'une forte stabilité entre les différentes versions ProDOS, mais il a fallu recourir à 3 adresses extérieures. Pour l'accrochage (\$9A8D pour les seuls I/O Hooks ou \$9A7E pour y ajouter la restauration du flag TRACE) et pour le décrochage de ProDOS, les adresses ne changent pas dans les versions 1.0, 1.0.1 et 1.0.2. Par contre, la routine fermant tous les fichiers s'est légèrement déplacée de \$B548 (1.0) à \$B54C (1.0.1 et 1.0.2). On sera donc prudent pour l'utilisation du programme proposé avec d'autres versions futures de ProDOS.

La théorie : DOUBLEDOS

Il se compose d'une partie d'initialisation réalisant tous les transferts nécessaires en mémoire auxiliaire, ainsi que la mise hors service du pseudo-disque /RAM et d'une partie "résidente" permettant de commuter le système d'exploitation par un seul et même CALL 768. Remarque : si l'on est perdu (3.3 ou Pro ?), un moyen extrêmement simple : faites RETURN en début de ligne ; ProDOS passe simplement à la ligne suivante, tandis que DOS 3.3 ajoute un retour chariot faisant ainsi sauter une ligne. Simple, mais efficace.

Le programme démarre en supposant que le DOS 3.3 a été chargé en mémoire ; le corps de 3.3 de \$4600 à \$6FFF et une

image de sa page 3 juste devant, de \$45D0 à \$45FF.

- \$2B1-\$2EE : c'est la routine (empruntée à Beneath Apple ProDOS) mettant hors service le volume /RAM, situé conventionnellement en slot 3, drive 2. On pourrait envisager de simplement protéger les zones concernées dans le VBM de /RAM à l'aide d'un appel à MLI, mais la correspondance numéro de bloc, page mémoire auxiliaire est loin d'être évidente : pour les blocs de \$D à \$5C, il faut retirer 8 au numéro de bloc, diviser le résultat par 17 et mettre les blocs dans la mémoire à partir de \$2000 vers

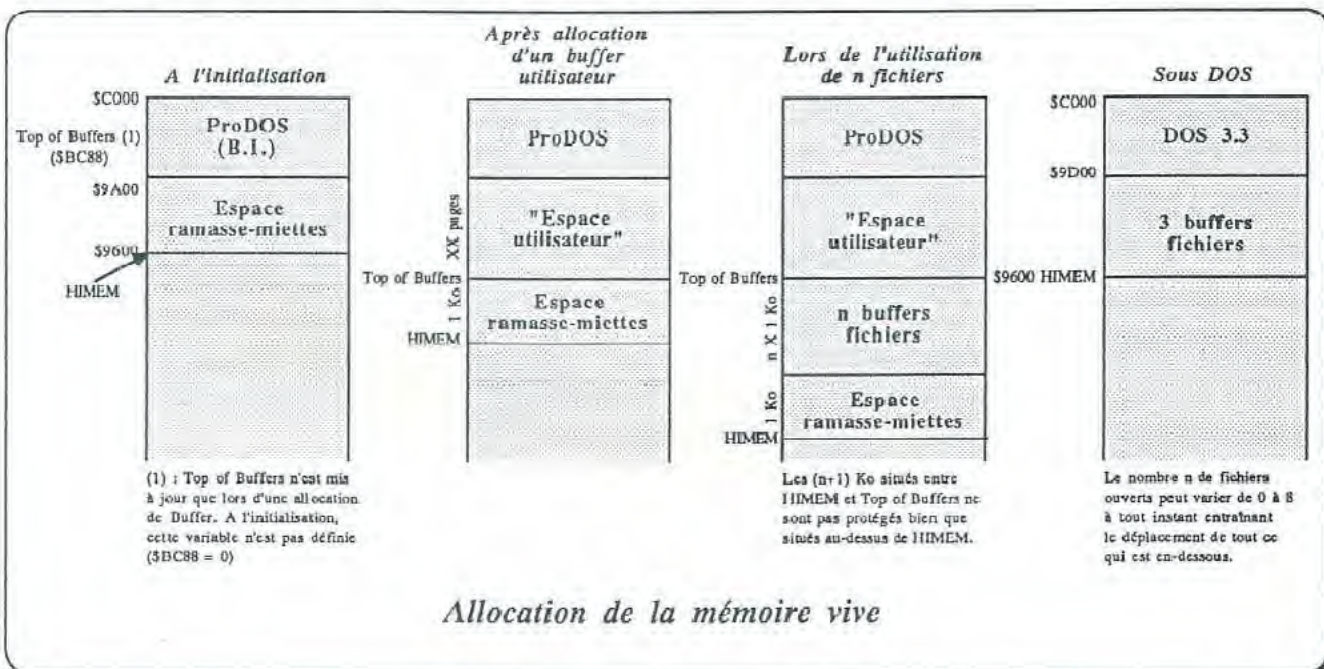
mémoire auxiliaire, de sa page 3 immédiatement en dessous et du bloc DOS et sa page 3 encore en dessous.

- \$300-\$304 : aiguillage du commutateur. On teste simplement le MSB du vecteur d'interruption masquable IRQ en \$3FE-\$3FF. Celui-ci vaut \$FF65 (entrée dans le moniteur) pour DOS 3.3 et \$BFE8 adresse du Handler d'interruption ProDOS, qu'il est déconseillé de modifier.

- les deux morceaux (Pro -> 3.3 et réciproquement) utilisent tous les deux les mêmes sous-programmes DOMOVES et

auxiliaire. On utilise une table des mouvements à effectuer en \$374-\$39E ; les 6 octets suivants \$39E ne sont utilisés qu'une fois lors de l'initialisation et deviennent donc disponibles immédiatement.

- \$30E-\$31F : avant d'appeler DOS, on ferme tous les fichiers pendents (à l'exception d'un éventuel fichier EXEC, il faut donc éviter de commuter dans un EXEC si l'on tient à ce qui se trouve après HIMEM) (voir remarque ci-dessus pour l'adresse de la routine CLOSEALL). Puis on s'enquiert de TOPBUFF en demandant



\$BFFF à l'exception de ceux dont le reste de la division par 17 vaut 1 qui vont successivement de \$12 à \$18. Et le reste à l'avenant : les blocs \$5D-\$5F vont en \$1A-\$1F, /RAM utilise les deux bancs \$D0-\$DF...

- \$2EF-\$2FF: transfert de ProDOS à la même adresse en

DOMOVE pour réaliser les transferts. Comme sa consœur MOVE du moniteur, AUXMOVE déplace une zone mémoire de (A1)-(A2) vers la zone débutant en (A4), le sens du mouvement est donné par le bit de retenue : s'il est mis à 1, le transfert s'effectue vers la mémoire

l'allocation de 0 page... En effet, la valeur de TOPBUFF est stockée en \$BC88 (donc hors pages globales alors que GETBUFR y est) mais il n'est mis à jour que lors d'allocation de buffers (fichiers ou utilisateurs) et pas lors de la libération. La routine GETBUFR retourne la

valeur correcte. Si TOPBUFF vaut \$9A, c'est que l'on a rien modifié, on ne touche donc à rien ; s'il est différent, c'est qu'il est inférieur ou égal à \$96, il faut donc remonter HIMEM DOS de 1K (4 pages) pour tenir compte du tampon ramasse miettes.

- \$332-\$344 : avant d'entrer dans ProDOS, on vérifie si le haut de la mémoire n'a pas été abaissé, dans ce cas, on alloue un buffer de taille suffisante pour faire descendre TOPBUFF à la hauteur du HIMEM DOS. Du coup HIMEM ProDOS est automatiquement plus petit de 1Ko que pour DOS. Remarquez que dès que l'on alloue quelque chose, on avale automatiquement les pages \$96 à \$9A, qui ne contiendront rien. C'est-à-dire, la première fois que l'on descend le haut de la mémoire, on le descend d'un minimum de 5 pages, y compris pour protéger un programme d'une vingtaine d'octets. Les autres fois on ne descend que du nécessaire.

La pratique

La suite des opérations est la suivante :

- Booter une disquette DOS :
- Faire :
 JCALL-151 *passé en moniteur*
 *45D0<3D0.3FFM *copie page 3*
 *4600<9600.BFFFM *copie Dos*
 *3DOG *retour au Basic*
 JBSAVE DOS, A\$45D0, L\$2A30 *sur une disquette 3.3*
- CONVERTir DOS sur une disquette ProDOS
- Assembler DOUBLEDOS (si l'on n'a qu'un assembleur DOS, on peut l'utiliser mais il faut évidemment de nouveau CONVERTir)
- Il ne reste plus qu'à enchaîner (éventuellement dans un

STARTUP) en faisant BLOAD DOS puis -DOUBLEDOS.0 pour se trouver en situation de commuter à tout moment par un simple : CALL 768.

Il peut être utile de commencer à utiliser cette commutation sur une disquette mixte ProDOS/DOS 3.3 et de sauvegarder la partie résidente du commutateur sur la partie DOS 3.3 de la disquette. En effet, la page 3 est quelque peu utilisée (euphémisme...) et vous risquez de la perdre et donc de ne plus pouvoir revenir sous ProDOS qu'en rebootant ou en réintroduisant le code machine à la main. Donc, dès le premier CALL768, faire :

SAVE DOUBLEDOS.OBJ, A\$300, L\$9E par exemple (on laissera tomber les tronçons d'initialisation).

On se convaincra de l'efficacité du système en expérimentant en mode direct et en entrant des programmes du genre :

```
10 PRINT CHR$(4) ; "CATA
LOG"
20 CALL768
30 PRINT CHR$(4) ; "CATA
LOG"
40 CALL768
```

ou encore pour utiliser APA, voir le programme LOADAPA. Apparemment, les diverses commandes APA, fonctionnent aussi sous ProDOS sauf AUTO (grande perte... et peut-être MERGE). Quelques remarques enfin, sur LOADAPA.

1.130 est indispensable car RBOOT vient en pages 2 et 3. 1.140 on se saisit du vecteur de l'Ampersand dans la page 3 DOS, et l'on POKE dans la page 3 ProDOS. (Il est d'ailleurs préférable de les POKer dans

XTRNADDR (\$BE50-51), Handler de commandes externes de ProDOS. Ceci lui permettrait ainsi de reconnaître également ces commandes. Evidemment, ici, ce n'est guère utile. Enfin, on notera qu'au premier passage de DOS vers ProDOS, après avoir chargé APA, vous pouvez perdre les valeurs de quelques variables alphanumériques, car HIMEM est arrondi à la page inférieure pour les besoins de ProDOS. Mais une fois cet ajustement effectué, vous pouvez passer sans problème d'un système à l'autre en conservant toutes vos variables, y compris en cours d'exécution.

Jonglez bien, et si vous êtes perdus, n'oubliez pas : un simple RETURN suffit.

NDLR : APA, logiciel commercial ne figure pas sur la disquette d'accompagnement.



Récapitulation

DOUBLEDOS.OBJ

```
0300- AE FF 03 E8 F0 24 20 00
0308- 9A AD 41 BE 85 F2 20 4C
0310- B5 A9 00 20 F5 BE C9 9A
0318- F0 07 18 A5 74 69 04 85
0320- 74 A2 17 20 49 03 20 51
0328- A8 60 20 E0 9E A2 29 20
0330- 49 03 A5 74 C9 96 F0 0D
0338- A9 00 85 73 20 F5 BE 38
0340- E5 74 20 F5 BE 20 7E 9A
0348- 60 38 20 5A 03 38 20 5A
0350- 03 18 20 5A 03 18 20 5A
0358- 03 60 08 A0 07 BD 74 03
0360- 99 3C 00 CA C0 06 D0 02
0368- 88 88 88 10 F0 28 EA 20
0370- 11 C3 AA 60 D0 6A FF 6A
0378- D0 03 00 6B FF 94 00 96
0380- D0 03 FF 03 D0 95 00 96
0388- FF BF 00 96 D0 95 FF 95
0390- D0 03 D0 03 FF 03 D0 6A
0398- 00 96 FF BF 00 6B D0
```

sur disquette DOS 3.3

Programme LOADAPA

```
100 CALL 768: REM ---- Je passe en 3.3
105 REM
110 PRINT CHR$(4) ; "BLOAD RBOOT"
120 CALL 520 :AD = USR (0) , "APA" : CALL AD
```

```
130 PRINT CHR$(4) ; "BLOAD DOUBLEDOS.OBJ"
140 LAMPV = PEEK (1014) :HAMPV = PEEK (1015)
150 CALL 768: REM ---- Je reviens en ProDOS
155 REM
160 POKE 1014,LAMPV: POKE 1015,HAMPV
170 PRINT : PRINT "Me revoilà en ProDOS"
180 NEW
```


Programme DOUBLEDOS (Assembleur Procode)

```

SOURCE  FILE #01 ==> /POMS/DOUBLEDOS
0000:      1      LST  NOA
0000:      2 *****
0000:      3 ; [change ProDOS <--> DOS
0000:      4 *****
----- NEXT OBJECT FILE NAME IS /POMS/DOUBLEDOS.0
02B1:      02B1  5      ORG  $2B1
02B1:      0073  6 HIMEM  EQU  $73
02B1:      003C  7 A1     EQU  $3C
02B1:      003E  8 A2     EQU  $3E
02B1:      0042  9 A4     EQU  $42
02B1:      00F2 10 TRACE  EQU  $F2
02B1:      A851 11 HOOKDOS EQU  $A851
02B1:      9EE0 12 UNHOOKDOS EQU $9EE0
02B1:      9A7E 13 HKPROTRCE EQU $9A7E
02B1:      9A8D 14 HOOKPRO  EQU  $9A8D
02B1:      9A00 15 UNHOOKPRO EQU  $9A00
02B1:      B54C 16 CLOSEALL EQU  $B54C
                        (V.1.0:$B548)
02B1:      BE0C 17 PRINTERR EQU $BE0C
02B1:      BE41 18 DTRACE  EQU  $BE41
02B1:      BEF5 19 GETBUFR  EQU  $BEF5
02B1:      BF16 20 DEVADR31 EQU  $BF16
02B1:      BF26 21 DEVADR32 EQU  $BF26
02B1:      BF31 22 DEVCNT  EQU  $BF31
02B1:      BF32 23 DEVLST  EQU  $BF32
02B1:      BF98 24 MACHID  EQU  $BF98
02B1:      C311 25 AUXMOVE  EQU  $C311
02B1:      26 ;
02B1:AD 98 BF      27      LDA  MACHID
02B4:29 30      28      AND  #$30
02B6:C9 30      29      CMP  #$30
02B8:F0 05      30      BEQ  OK128
02BA:A9 03      31 NORAM   LDA  #3
                        'Device not connected'
02BC:4C 0C BE      32      JMP  PRINTERR
02BF:AD 16 BF      33 OK128 LDA  DEVADR31
02C2:CD 26 BF      34      CMP  DEVADR32
02C5:F0 28      35      BEQ  TRANSF
02C7:8D 26 BF      36      STA  DEVADR32
02CA:AD 17 BF      37      LDA  DEVADR31+1
02CD:8D 27 BF      38      STA  DEVADR32+1
02D0:AE 31 BF      39      LDX  DEVCNT
02D3:BD 32 BF      40 DEVLPLDA  DEVLST,X
02D6:29 70      41      AND  #$70
02D8:C9 30      42      CMP  #$30
02DA:F0 05      43      BEQ  GOTSILT
02DC:CA      44      DEX
02DD:10 F4      45      BPL  DEVLPL
02DF:30 D9      46      RMT  NORAM
02E1:BD 33 BF      47 GOTSILT LDA  DEVLST+1,X
02E4:9D 32 BF      48      STA  DEVLST,X
02E7:F0 03      49      BEQ  REMOVE
02E9:E8      50      INX
02EA:D0 F5      51      BNE  GOTSILT
02EC:CE 31 BF      52 REMOVE DEC  DEVCNT
02EF:      53 ; initial transfert --> /RAM
02EF:A2 2F      54 TRANSF LDX  #47
02F1:38      55      SEC
02F2:20 5A 03      56      JSR  DOMOVE
02F5:A2 17      57      LDX  #23
02F7:38      58      SEC
02F8:20 5A 03      59      JSR  DOMOVE
02FB:38      60      SEC
02FC:20 5A 03      61      JSR  DOMOVE
02FF:60      62      RTS
0300:      63 ;
0300:      64 ; partie résidente
0300:AE FF 03      65 ENTREE LDX  $3FF
0303:E8      66      INX
0304:F0 24      032A 67      BEQ  CALLPRO
0306:      68 ;
0306:20 00 9A      69 CALLDOS JSR  UNHOOKPRO
0309:AD 41 BE      70      LDA  DTRACE
030C:85 F2      71      STA  TRACE
030E:120 4C B5      72      JSR  CLOSEALL
0311:A9 00      73      LDA  #0
0313:20 F5 BE      74      JSR  GETBUFR
0316:C9 9A      75      CMP  #$9A
0318:F0 07      0321 76      BEQ  CLDOS1
031A:18      77      CLC
031B:A5 74      78      LDA  HIMEM+1
031D:69 04      79      ADC  #4
031F:85 74      80      STA  HIMEM+1
0321:A2 17      81 CLDOS1 LDX  #23
0323:20 49 03      82      JSR  DOMOVES
0326:20 51 A8      83      JSR  HOOKDOS
0329:60      84      RTS
032A:      85 ;
032A:20 E0 9E      86 CALLPRO JSR  UNHOOKDOS
032D:A2 29      87      LDX  #41
032F:20 49 03      88      JSR  DOMOVES
0332:A5 74      89      LDA  HIMEM+1
0334:C9 96      90      CMP  #$96
0336:F0 0D      0345 91      BEQ  CLPRO1
0338:A9 00      92      LDA  #0
033A:85 73      93      STA  HIMEM
033C:20 F5 BE      94      JSR  GETBUFR
033F:38      95      SEC
0340:E5 74      96      SBC  HIMEM+1
0342:20 F5 BE      97      JSR  GETBUFR
0345:20 7E 9A      98 CLPRO1 JSR  HKPROTRCE
0348:60      99      RTS
0349:      100 ;
0349:38      101 DOMOVES SEC
034A:20 5A 03      102      JSR  DOMOVE
034D:38      103      SEC
034E:20 5A 03      104      JSR  DOMOVE
0351:18      105      CLC
0352:20 5A 03      106      JSR  DOMOVE
0355:18      107      CLC
0356:20 5A 03      108      JSR  DOMOVE
0359:60      109      RTS
035A:      110 ; appel AUXMOVE transfert
                        n*(X+1)/6
035A:08      111 DOMOVE  PHP
035B:A0 07      112      LDY  #7
035D:BD 74 03      113 DMV1  LDA  CMDTBL,X
0360:99 3C 00      114      STA  A1,Y
0363:CA      115      DEX
0364:C0 06      116      CPY  #6
0366:D0 02      036A 117      BNE  DMV2
0368:88      118      DEY
0369:88      119      DEY
036A:88      120 DMV2  DEY
036B:10 F0      035D 121      BPL  DMV1
036D:28      122      PLP
036E:8A      123      TXA
036F:20 11 C3      124      JSR  AUXMOVE
0372:AA      125      TAX
0373:60      126      RTS
0374:      127 ; arguments de Move OS --> /RAM
0374:D0 6A      128 CMDTBL DW  $6AD0,$6AFF,$03D0
037A:00 6B      129      DW  $6B00,$94FF,$9600
0380:D0 03      130      DW  $03D0,$03FF,$95D0
0386:      131 ;
0386:00 96      132      DW  $9600,$BFFF,$9600
038C:      133 ;
038C:D0 95      134      DW  $95D0,$95FF,$03D0
0392:D0 03      135      DW  $03D0,$03FF,$6AD0
0398:00 96      136      DW  $9600,$BFFF,$6B00
039E:      039E 137 END  EQU  *
039E:D0 45      138      DW
$45D0,$6FFF,$6AD0

```


Copie d'écran graphique

On peut se demander la raison d'être d'un tel programme puisque la ROM de la carte d'interface de Silentype comprend déjà une routine de copie d'écran graphique activée par CTRL-Q. Malheureusement, l'image ainsi obtenue m'a toujours paru trop petite et c'est ce qui m'a amené à écrire ce programme qui fournit une image quatre fois plus grande que celle obtenue par CTRL-Q, soit environ un format A4. Bien entendu, il a fallu renverser l'image qui ne sera plus imprimée ligne par ligne mais colonne par colonne.

Le programme Basic n'appelle pas de commentaires particuliers. Il se contente de proposer un menu permettant, à l'utilisateur, de choisir les paramètres de l'imprimante et la page graphique 1 ou 2.

Noter au passage à partir de 2000 la routine de lecture du catalogue en Basic (seul l'appel de RWTS est en langage machine et n'occupe que 6 octets). Les noms de fichiers binaires sont transférés dans le tableau F\$. Le programme ne vérifie pas la longueur de ces fichiers, il est donc de la responsabilité de l'utilisateur de s'assurer qu'il s'agit bien d'une page graphique. Le contenu des pages graphiques peut être vérifié en appuyant sur les touches 1 ou 2.

La routine en langage machine RECOP.OBJ est cachée à la fin du programme Basic. Elle est entièrement relogable ce qui permet de modifier ce dernier à sa guise. Son rôle est de dédoubler chaque colonne de la page graphique avant d'en imprimer chaque octet deux fois de suite. Ainsi, un point à l'écran correspondra à quatre points sur le papier. L'impression peut être interrompue en appuyant sur la touche ESC, tandis que le mode d'impression (uni ou bidirectionnel) peut être modifié en cours d'impression par les touches U ou B.

Enfin, si vous n'avez pas la disquette d'accompagnement, pour recopier le programme, il faut dans l'ordre suivant :

- recopier le programme Basic sans en changer un octet ;
- vérifier que le pointeur de fin de programme (\$AF - \$B0) indique une valeur inférieure à \$1573. Modifier ce pointeur en le mettant à \$16BC ;
- recopier le programme en langage machine à partir de \$1573 et sauvegarder le programme Basic ainsi obtenu.



Programme RECOP.OBJ

1573- A9 01 20 95 FE

1578- 20 8E FD A9 20 20 AB CC
1580- A9 11 85 E0 A9 01 85 E1
1588- AD 13 CF 85 E6 A9 00 85
1590- 2A 85 E2 85 CE 85 CF A9
1598- 1E 85 2B A5 E2 A6 E0 A4
15A0- F1 20 11 F4 A2 02 B1 26
15A8- 4A 08 66 CE 28 66 CE CA
15B0- 10 F6 4A 48 08 66 CE A4
15B8- E2 A5 CE 4A 4D 14 CF F0
15C0- 04 A2 FF 86 CF 91 2A A9
15C8- 00 85 CE A9 1F 85 2B 28
15D0- 66 CE A2 02 68 4A 08 66
15D8- CE 28 66 CE CA 10 F6 A5
15E0- CE 4A 4D 14 CF F0 04 A2
15E8- FF 86 CF 91 2A A9 00 85
15F0- CE E6 E2 A5 E2 C9 C0 90
15F8- 9E 24 CF 30 0C A9 08 20
1600- AB CC 18 90 38 A9 00 10
1608- 88 A2 02 86 CF A0 00 8C
1610- 01 CF B1 2A 8D 2B CF 20
1618- 0B CB 20 A4 CB 20 0B CB
1620- 20 A4 CB C8 C0 C0 90 EA
1628- A9 04 20 AB CC A9 1E 85
1630- 2B 2C 0F CF 10 36 20 02
1638- CD C6 CF D0 D0 AD 00 C0
1640- 10 1B C9 9B F0 62 C9 D5
1648- D0 07 A9 FF 8D 0F CF D0
1650- 09 C9 C2 D0 05 A9 00 8D
1658- 0F CF 2C 10 C0 A5 E0 38
1660- E9 07 85 E0 B0 9F C6 E1
1668- 10 9B 30 3F A0 FF 8C 01
1670- CF A9 00 8D 2B CF 20 0B
1678- CB 20 A4 CB 20 0B CB 20
1680- A4 CB 20 0B CB A0 C0 88
1688- 08 B1 2A 8D 2B CF 20 0B
1690- CB 20 A4 CB 20 0B CB 20
1698- A4 CB 28 D0 EA A9 04 20
16A0- AB CC 20 02 CD 18 90 95
16A8- 2C 10 C0 A9 1E 20 AB CC
16B0- 4C 93 FE 20 E3 03 4C D9
16B8- 03 00 00 00 00

Programme RECOP.SOURCE (Assembleur Toolkit)

```

1      ORG $1573
2      *
3      *****
4      ***** RECOPIE D'ECRAN GRAPHIQUE SUR SILENTYPE *****
5      *****
6      ***** Auteur :  BECHER GERARD   10/06/84   *****
7      *****
8      *****
9      *
10     *****
11     HBASL EQU $26      ;Adresse d'un octet de la page HGR
12     IMAGL EQU $2A      ;Adresse de l'image agrandie

```

```

13     IMAGH EQU $2B      ;d'une colonne
14     TEMP EQU $CB       ;Reçoit l'image agrandie d'un octet
15     TEMOIN EQU $CF     ;Zéro si une ligne est vide.
16     NC EQU $E0         ;Numéro de la colonne courante
17     NL EQU $E2         ;Numéro de la ligne courante
18     HPAGE EQU $E6      ;Page HGR courante
19     KBD EQU $C000      ;Adresse du clavier
20     KBDSTR EQU $C010   ;Strobe du clavier
21     IMPRIME EQU $CB08   ;Imprime une barre verticale.
22     BARRE EQU $CF2B     ;Octet imprimé par IMPRIME
23     TEMPO EQU $CBA4     ;Temporisation
24     SENS EQU $CF01      ;Détermine le sens d'impression
25     DIR EQU $CF0F       ;Uni ou bidirectionnel?
26     PAGE EQU $CF13      ;Page Haute résolution ($20 ou $40)
27     INVERSE EQU $CF14   ;Mode normal ou inverse.
28     DEROULE EQU $CCAR   ;Déroulement du papier
29     RETOUR EQU $CD02     ;Retour de la tête en colonne 0.

```


sur Silentype

Gérard Becher

```
30 HPOEN EQU $F411 ;Calcule l'adresse d'un point HCR
31 CROUT EQU $FD8E ;Envoie un Retour Chariot
32 SETVID EQU $FE93 ;Simule PR#
33 *****
34 *
35 LDA #1
36 JSR SETVID+2 ;PR#1
37 JSR CROUT ;Envoie un Retour Chariot
38 LDA #520
39 JSR DEROULE ;Déroule le papier
40 LDA #>273 ;Initialise le numéro de colonne
41 STA NC ;(commence par la dernière colonne)
42 LDA #<273
43 STA NC+1
44 LDA PAGE ;Fixe le numéro de page pour HPOEN
45 STA HPAGE
46 LDA #0
47 STA IMACL
48 DOUBLE STA NL ;Initialise le numéro de ligne
49 STA TEMP
50 STA TEMOIN ;(Reste à 0 si la colonne est vide)
51 SUIVANT LDA #S1E ;La 1ère série de 192 octets sera
52 STA IMAGH ;rangée en $1E00, la 2ème en $1F00
53 LDA NL ;Calcule l'adresse de base d'un octet
54 LDX NC ;de la page graphique en fonction de
55 LDY NC+1 ;son numéro de ligne et de colonne.
56 JSR HPOEN
57 LDX #2 ;Décale chaque octet bit par bit
58 LDA (HBASL),Y ;en dédoublant chacun des sept bits
59 DECAL1 LSR A ;L'octet est décalé à droite et
60 PHP ;chaque bit qui tombe dans la retenue
61 ROR TEMP ;sera réintégré deux fois dans
62 PLP ;l'octet de travail temporaire TEMP
63 ROR TEMP
64 DEX ;On répète ce processus trois fois
65 BPL DECAL1 ;soit six bits dans TEMP
66 LSR A ;Le bit suivant tombe dans la retenue
67 PHA ;et formera le septième bit de TEMP
68 PHP ;Sauvegarde de ce bit qui n' a été
69 ROR TEMP ;réintégré qu'une fois dans TEMP
70 LDY NL
71 LDA TEMP ;L'octet résultant est chargé
72 LSK A ;son bit de poids fort mis à zéro
73 EOR INVERSE ;Eventuellement il est inversé
74 BEQ NUL1 ;Un octet nul ne modifie pas TEMOIN
75 LDX #SFF ;tandis qu'un octet non nul le mettra
76 STX TEMOIN ;irrémodifiablement à SFF
77 NUL1 STA (IMAGL),Y ;Résultat rangé à partir de $1E00
78 LDA #0 ;La mémoire de travail TEMP est
79 STA TEMP ;Remise à zéro
80 LDA #S1F ;La prochaine série sera rangée
81 STA IMAGH ;en $1F00 au lieu de $1E00
82 PLP ;On récupère le bit qui avait été
83 ROR TEMP ;sauvegardé et on le réintègre
84 LDX #2 ;dans TEMP. Les trois bits restants
85 PLA ;sont dédoublés de la même façon
86 DECAL2 LSR A ;que précédemment
87 PHP ;et formeront un nouvel octet TEMP
88 ROR TEMP ;qui sera traité comme le premier
89 PLP
90 ROR TEMP
91 DEX
92 BPL DECAL2
93 LDA TEMP ;puis rangé à partir de $1F00
94 LSR A ;bit 7 à zéro
95 EOR INVERSE ;Eventuellement inversé
96 BEQ NUL2
97 LDX #SFF ;TEMOIN mis à SFF car
98 STX TEMOIN ;l'octet en question n'est pas nul.
99 NUL2 STA (IMAGL),Y ;Rangement du résultat
100 LDA #0
101 STA TEMP ;Remise à 0 de la mémoire de travail
102 INC NL ;Comptage des octets de la colonne.
103 LDA NL ;Les 192 octets de la colonne
104 CMP #192 ;ont-ils été tous traités ?
105 BCC SUIVANT ;NON => Octet suivant
106 *
107 *****
108 * Impression des deux séries de 192 octets

109 *****
110 *
111 BIT TEMOIN ;La ligne à imprimer est-elle vide?
112 DMI IMPLIN ;NON -> Va l' imprimer
113 LDA #8 ;OUI =>
114 JSR DEROULE ;Se contente de dérouler le papier
115 CLC
116 BCC CLAVIER
117 *
118 *****
119 *
120 SAUT2 LDA #0
121 BPL DOUBLE ;Relais pour remonter jusqu'à DOUBLE
122 *
123 *****
124 *
125 IMPLIN LDX #2
126 STX TEMOIN
127 LINDIR2 LDY #0 ;Sélectionne le sens
128 STY SENS ;de gauche à droite
129 LINDIR1 LDA (IMAGL),Y ;Chaque octet de la série est imprimé
130 STA BARRE ;sous la forme d'une double barre
131 JSR IMPRIME ;Imprime une barre verticale
132 JSR TEMPO ;Temporisation
133 JSR IMPRIME ;Double la barre imprimée.
134 JSR TEMPO
135 INY ;Octet suivant
136 CPY #192 ;Dernier octet ?
137 BCC LINDIR1 ;NON => On recommence
138 LDA #4 ;OUI => Avance le papier (LF)
139 JSR DEROULE
140 LDA #S1E ;Pointe vers la 1ère série d'octets
141 STA IMAGH
142 BIT DIR ;Bi- ou Uni-directionnel ?
143 BPL BIDIR ;BI
144 JSR RETOUR ;UNI -> revient en début de ligne
145 DEC TEMOIN ;et effectue la 2ème passe
146 BNE LINDIR2
147 *
148 *****
149 *
150 CLAVIER LDA KBD ;Test du clavier.
151 BPL SUITE2 ;Pas de touche enfoncée.
152 CMP #C9B ;ESCAPE ?
153 BEQ ARRET ;Oui => Arrêt impression.
154 CMP #U' ;U ?
155 BNE CLAV1
156 LDA #SFF ;=> Mode Uni-directionnel.
157 STA DIR
158 RNF SUITE
159 CLAV1 CMP #B' ;B ?
160 DNE SUITE
161 LDA #0 ;=> Mode Bi-directionnel
162 STA DIR
163 SUITE BIT KBDSTR ;Efface le clavier
164 SUITE2 LDA NC ;Passes à la colonne précédente
165 SEC ;en reculant de 7 points
166 SBC #7
167 STA NC
168 BCS SAUT2
169 DEC NC+1
170 BPL SAUT2 ;Et remonte jusqu'en DOUBLE
171 BMI FIN ;Terminé !
172 *
173 *****
174 *
175 BIDIR LDY #SFF ;Change le sens d'impression
176 STY SENS ;(de droite à gauche)
177 LDA #0
178 STA BARRE ;Rattrape un décalage de 3 barres
179 JSR IMPRIME
180 JSR TEMPO
181 JSR IMPRIME
182 JSR TEMPO
```



```

183 JSR IMPRIME
184 LDY #192 ; Imprime la ligne à l'envers
185 LININV DEY ; en commençant par le dernier octet
186 PHP
187 LDA (IMAGL),Y
188 STA BARRE
189 JSR IMPRIME ; imprime une barre verticale
190 JSR TEMPO ; Temporisation
191 JSR IMPRIME ; Double la barre imprimée.
192 JSR TEMPO
193 PLP ; Terminé ? (Y = 0?)
194 BNE LININV ; NON => Octet précédent
195 LDA #4 ; OUI -> Line Feed
196 JSR DEROULE ; (Descend d'un cran)
197 JSR RETOUR ; Et se repositionne en début de ligne
198 CLC

```

```

199 BCC CLAVIER
200 *
201 *****
202 *
203 ARRET BIT KBDSTR
204 FIN LDA #30
205 JSR DEROULE ; Deroule le papier.
206 JMP SETVID ; PR#0
207 *
208 *****
209 *
210 JSR $3E3 ; Appel de RWTS
211 JMP $3D9
212 BRK
213 BRK
214 BRK

```

Programme RECOPECRAN

NB : A la suite de ce programme est "cachée" une routine en langage machine. Voir texte.

```

10 TEXT : HOME
20 PRINT "RECOPIE D'ECRAN GRAPHIQUE SUR SILENTYPE"
30 PRINT "-----"
40 LOMEM: 24576
50 POKE 34,2: POKE 33,33: POKE 32,7
60 OF = 0: A$ = "": AD = VAL(A$) + PEEK(131) + 256 * PEEK(132): I = 0
70 MG = 10: IN = 7: PG = 1
80 DIM F$(105)
90 REM Impression des valeurs par défaut
100 HOME : NORMAL
110 VTAB 6: PRINT "CHARGEMENT IMAGE ": GOSUB 1100
120 VTAB 8: PRINT "MARGE GAUCHE ": GOSUB 1200
130 VTAB 10: PRINT "INTENSITE ": GOSUB 1300
140 VTAB 12: PRINT "PAGE ": GOSUB 1400
150 GOSUB 1500: GOSUB 1600
160 VTAB 21: HTAB 1: CALL - 958: PRINT " <ESPACE> POUR MODIFIER": PRINT "<RETURN> POUR POURSUIVRE": PRINT "<ESC> POUR QUITTER": PRINT "<1> OU <2> POUR HGR":
170 GOSUB 1000: IF A = 141 THEN 600
180 IF A = 155 THEN TEXT : HOME : END
185 GOSUB 3000
190 IF A < > 160 THEN 170
200 REM Modification des paramètres
210 VTAB 21: HTAB 1: CALL - 958: PRINT "<ESPACE> POUR CHOISIR": PRINT "<1> OU <2> POUR MODIFIER": PRINT "<RETURN> POUR CONCLURE": PRINT "<1> OU <2> POUR HGR":
220 IT = 0
230 ON IT GOTO 300,350,400,450,500
250 INVERSE : GOSUB 1110: GOSUB 1000: IF A = 160 THEN IT = 1: GOSUB 1100: GOTO 230
260 IF A = 149 THEN IM = (IM + 1) * (IM < 2): GOTO 250
270 IF A = 136 THEN IM = (IM - 1) * (IM > 0) + 2 * NOT IM: GOTO 250
280 IF A = 141 OR A = 155 THEN 100
290 GOSUB 3000: GOTO 250
300 INVERSE : GOSUB 1210: GOSUB 1000: IF A = 160 THEN IT = 2: GOSUB 1200: GOTO 230
310 IF A = 149 THEN MG = (MG + 1) * (MG < 20): GOTO 300
320 IF A = 136 THEN MG = (MG - 1) * (MG > 0) + 21 * NOT MG: GOTO 300
330 IF A = 141 OR A = 155 THEN 100
340 GOSUB 3000: GOTO 300
350 INVERSE : GOSUB 1310: GOSUB 1000: IF A = 160 THEN IT = 3: GOSUB 1300: GOTO 230
360 IF A = 149 THEN IN = (IN + 1) * (IN < 7): GOTO 350
370 IF A = 136 THEN IN = (IN - 1) * (IN > 0) + 7 * NOT IN: GOTO 350
380 IF A = 141 OR A = 155 THEN 100
390 GOSUB 3000: GOTO 350
400 INVERSE : GOSUB 1410: GOSUB 1000: IF A = 160 THEN IT = 4: GOSUB 1400: GOTO 230

```

```

410 IF A = 149 OR A = 136 THEN PG = NOT (PG - 1) + 1: GOTO 400
420 IF A = 141 OR A = 155 THEN 100
430 GOSUB 3000: GOTO 400
450 INVERSE : GOSUB 1510: GOSUB 1000: IF A = 160 THEN IT = 5: GOSUB 1500: GOTO 230
460 IF A = 149 OR A = 136 THEN MOD = NOT MOD: GOTO 450
470 IF A = 141 OR A = 155 THEN 100
480 GOSUB 3000: GOTO 450
500 INVERSE : GOSUB 1610: GOSUB 1000: IF A = 160 THEN IT = 0: GOSUB 1600: GOTO 230
510 IF A = 149 OR A = 136 THEN DIR = NOT DIR: GOTO 500
520 IF A = 141 OR A = 155 THEN 100
530 GOSUB 3000: GOTO 500
600 IF NOT IM THEN 700: REM Pas d'image à charger
610 GOSUB 2000: REM Lecture catalogue
620 IF NOT I THEN 100: REM Pas d'image sur ce disque.
630 VTAB 22: HTAB 1: PRINT "NOM DE FICHIER, NUMERO, OU <RETURN>":
640 INPUT "==" : A$: IF LEN(A$) = 0 THEN 100
650 IF VAL(A$) > 0 AND VAL(A$) < -1 THEN A$ = F$(VAL(A$) - 1)
660 PRINT CHR$(4) "BLOAD": A$: "A": 8192 * PG: "D": IM
670 IM = 0: GOTO 100
700 REM On initialise l'imprimante
710 REM Et on fixe les paramètres
720 PRINT : PRINT CHR$(4) "PR#1": PRINT
730 PRINT CHR$(4) "PR#0"
740 POKE - 12527, MG: POKE - 12528, IN: POKE - 12529, 255 * DIR: POKE - 12524, 255 * MOD: POKE - 12525, 32 * PG
745 POKE 49239, 0: POKE 49234, 0: POKE 49235 + PG, 0: POKE 49232, 0
750 CALL PEEK(175) + 256 * PEEK(176) - 329: GOSUB 3030: GOTO 100
1000 IF PEEK(49152) < 128 THEN 1000
1010 A = PEEK(49152): POKE 49168, 0: RETURN
1100 NORMAL
1110 VTAB 6: HTAB 20: CALL - 868: ON IM GOTO 1130, 1140
1120 PRINT "NON": RETURN
1130 PRINT "DRIVE 1": RETURN
1140 PRINT "DRIVE 2": RETURN
1200 NORMAL
1210 VTAB 8: HTAB 16: CALL - 868: PRINT MG: RETURN
1300 NORMAL
1310 VTAB 10: HTAB 16: CALL - 868: PRINT IN: RETURN
1400 NORMAL
1410 VTAB 12: HTAB 16: CALL - 868: PRINT PG: RETURN
1500 NORMAL
1510 VTAB 14: CALL - 868: PRINT "MODE ": IF MOD THEN PRINT "INVERSE": RETURN
1520 PRINT "NORMAL": RETURN
1600 NORMAL
1610 VTAB 16: CALL - 868: IF DIR THEN PRINT "UNI": GOTO 1630
1620 PRINT "BI":

```

Suite de Recopecran page suivante =>

Manipulation de catalogue

Jean-Luc Arnaud

Ce programme, écrit avec l'assembleur Lisa 1.5, vous permet de modifier le catalogue de votre disquette de façon simple et efficace. Ainsi, vous pouvez renommer un fichier, déplacer une zone de fichiers, échanger la place de deux fichiers, ...

Mode d'emploi

Lors du chargement du programme, l'écran se divise en deux fenêtres :

- une fenêtre menu sur le bas de l'écran, sur laquelle s'afficheront les différentes fonctions auxquelles vous aurez accès ;
- une fenêtre de travail sur le reste de l'écran.

En premier lieu, vous devez demander le chargement d'un catalogue. La première option à choisir est donc "C". Le programme vous demande alors la confirmation de votre choix ; quatre réponses sont autorisées :

- O pour Oui ;
- N pour Non ;
- 1 ou 2 correspond au numéro du lecteur où se trouve la disquette dont le catalogue doit

être manipulé.

Le catalogue de votre disquette étant chargé, le programme vous propose :

- la lecture
- l'édition
- la modification
- le réenregistrement
- la sortie du programme.

Lecture

Ce choix est l'équivalent d'un CATALOG, il vous permet donc de visualiser les noms de fichiers de votre catalogue. En *LECTURE* le programme vous demande le numéro du premier secteur et le nombre de secteurs à lire (il vérifie les dépassements). Les titres s'affichent alors et sont précédés d'un numéro (1 à 255, voir explications techniques) et du signe +, - ou &, suivant qu'il s'agit d'un fichier existant (+), d'un fichier effacé (-) ou d'un commentaire (&).

Signalons à ce propos que lors d'un DELETE, le fichier n'est pas effacé de la disquette. Le DOS se contente de remplacer le numéro de piste de la liste pistes/secteurs du fichier concer-

né par \$FF : on peut donc toujours retrouver dans le catalogue un nom de fichier effacé auparavant.

Lors de l'affichage du catalogue, si le nombre de fichiers est supérieur au nombre de lignes disponibles dans la fenêtre, la touche ESPACE vous permettra de continuer et de visualiser tous les noms des différents fichiers composant le catalogue. Dans ce cas, comme à l'entrée des informations, le nombre de secteurs occupés est inscrit. Cette opération effectuée, le programme vous renvoie au sous-menu *LECTURE*.

Edition

Le choix *EDITION* conduit à un sous-menu qui propose l'échange entre deux titres ou le déplacement d'une zone.

L'ECHANGE ne nécessite pas d'explications particulières. Il suffit de donner les numéros des deux titres à échanger ; ainsi, l'ordre des noms de fichiers est modifié. Le programme renvoie ensuite à l'écran les titres du secteur - catalogue déterminé par le 2ème numéro.

Suite de RECOPECRAN

```
1630 PRINT "DIRECTIONNEL": RETURN
2000 REM Sous progr. de lecture du catalogue
2010 HOME
2020 POKE 47082,10: REM Drive
2030 POKE 47083,0: REM Volume
2040 POKE 47084,17: REM Piste
2050 POKE 47088,187: POKE 47089,180: REM Buffer en $B4B8
2060 POKE 47092,1: REM Read
2070 I = 0: S = 0: P = 0
2080 POKE 47085,15: S: REM Secteur
2090 CALL PEEK(175) + 256 * PEEK(176) - 9
2100 IF PEEK(47093) THEN PRINT: PRINT CHR$(7); "/O
      ERROR": POP: TEXT: END
2110 OF = 11: POKE AD,30
2120 IF PEEK(46267 + OF) = 255 THEN 2180
2130 IF PEEK(46267 + OF) = 0 THEN
      2210: REM Fin du catalogue
```

```
2140 A = PEEK(46269 + OF): IF NOT (A = 4 OR A = 132) THEN
      2180
2150 POKE AD + 2, INT ((46270 + OF) / 256): POKE AD + 1, 46270
      + OF - 256 * PEEK(AD + 2): F$(I) = A$
2160 IF I AND 1 = 18 * INT (I / 18) THEN VTAB 23: PRINT
      "APPUYEZ SUR UNE TOUCHE": PRINT "POUR
      CONTINUER": GOSUB 1000: HOME
2170 VTAB 4 + I - 18 * INT (I / 18): I = I + 1: PRINT I: HTAB 4:
      PRINT A$
2180 OF = OF + 35
2190 IF OF < 255 THEN 2120
2200 S = S + 1: GOTO 2080
2210 RETURN
3000 REM VOIR UNE PAGE HGR
3005 IF NOT (A = 177 OR A = 178) THEN RETURN
3010 POKE 49239,0: POKE 49234,0: POKE 49236 + (A = 178),0:
      POKE 49232,0
3020 FOR I = 0 TO 1000: NEXT
3030 POKE 49236,0: POKE 49233,0: RETURN
```


Exemple :

A partir du menu général, tapez E (Edition) puis E (Echange) et enfin 3,40. Ainsi, les titres n°3 et 40 sont échangés et le secteur 6 est affiché (titres 36 à 42). De la même manière, on aurait pu entrer 40,3 ; dans ce cas, c'est le secteur 1 (titres 1 à 7) qui s'affiche.

Le **DEPLACEMENT** permet d'insérer des titres ou des commentaires rapidement. Il suffit d'entrer les limites supérieure et inférieure de la zone à traiter et le numéro du titre à partir duquel la zone sera implantée. Il est évident qu'il s'agit, en fait, d'un échange entre la zone à déplacer et la zone de destination. Le programme vérifie que le transfert ne se fait pas au-delà des limites du catalogue.

Exemple :

A partir du menu général, tapez E (Edition) puis D (Déplacement) et entrez les valeurs 1,10,40. La zone 1 à 10 est déplacée en 40 à 49 tandis que la zone 40-49 passe en 1-10.

Si les valeurs entrées sont 1,10,2 : la zone 1-10 passe en 2-11, 11 passe en 1 : il y a insertion d'une ligne.

On peut aussi faire "remonter" une zone :

Exemple :

Supposons que les valeurs saisies sont 90,100,30 : la zone 90-100

passera en 30-40 et vice-versa. Ainsi, les entrées 90,100,30 et 100,90,30 sont équivalentes.

Modification

Choisissons maintenant la **MODIFICATION** d'un titre. Cette partie du programme offre les possibilités suivantes :

- modification du titre d'un fichier ;
- mise en place de commentaires (texte et symboles divers) ;
- effacement du titre d'un fichier précédemment **DELETÉ** ou d'un commentaire. Par précaution, il est impossible d'effacer le titre d'un fichier présent.

Afin de modifier le nom d'un fichier, il faut entrer le numéro du titre à traiter. Celui-ci s'affiche précédé de son numéro, et surmonté d'une échelle, de 30 caractères, graduée de 1 en 1, repérée de 5 en 5. Le curseur est positionné sur le premier caractère du titre.

Les facilités de l'éditeur (touches de déplacement du curseur et fonction **ESCAPE**) sont naturellement accessibles pour cette modification.

Le programme détermine lui-même si vous manipulez un commentaire, un nom de fichier présent ou un titre de fichier déjà effacé. La seule précaution à prendre est de ne pas remplacer un titre de programme par une suite de symboles non acceptée par le DOS, celui-ci ne le

retrouverait plus. Il est donc indispensable de redonner un nom reconnaissable par le DOS.

L'effacement automatique s'obtient en tapant un **RETURN** à vide lors de l'entrée du texte. Après confirmation et contrôle, le texte est effacé. Il est également possible de taper **ESPACE** et **RETURN** ; dans ce cas le texte est effacé et remplacé par un 'blanc' qui apparaît lors du catalogue, sous forme d'une ligne vide (pour aérer...). Une tentative d'effacement du titre d'un fichier présent conduit à l'affichage d'un message d'erreur, avec retour au menu général.

Réenregistrement

Votre œuvre mise au point, il s'agit de la sauvegarder en la réenregistrant sur la **MEME** disquette. Choisissez simplement l'option **R** (Réenregistrement) qui transfère le contenu de la mémoire vers la disquette. Cette option est éminemment dangereuse puisqu'elle détruit le catalogue original et le remplace par celui que vous avez préparé (croyez-en mes malheureuses expériences...). Il est donc nécessaire de confirmer le choix par **1/2/O/N** : lecteur 1 ou 2, Oui, Non.

Attention : il est indispensable d'enregistrer le catalogue sur la **MEME** disquette. Sinon adieu vos beaux fichiers !

Terminer

Vous pouvez recommencer une

Récapitulation LECTCAT

0801- A9 4C 8D F5 03 A9 10
0808- 8D F6 03 A9 08 8D F7 03
0810- A2 00 86 00 B5 00 9D 00
0818- 78 CA D0 F8 A9 F0 85 36
0820- A9 FD 85 37 A9 1B 85 38
0828- A9 FD 85 39 20 58 FC A9
0830- 12 85 25 4C A5 08 42 08
0838- FE 09 52 0B 07 0D 87 0E
0840- 13 09 20 5B 10 20 C6 10
0848- 20 62 FC A9 00 85 00 85
0850- 18 A9 11 8D B5 10 A9 0F
0858- 8D B6 10 A9 20 8D BA 10
0860- A9 01 8D BD 10 A9 01 85
0868- CE A9 1F 85 CF 20 A9 10

0870- E6 CF A9 00 A8 D1 CE D0
0878- 17 C8 D1 CE D0 12 A5 18
0880- 18 69 01 0A 0A 0A 38 E5
0888- 18 85 06 C6 06 4C A5 08
0890- A0 00 B1 CE 8D B5 10 C8
0898- B1 CE 8D B6 10 EE BA 10
08A0- E6 18 4C 6D 08 A5 25 85
08A8- 08 A9 00 85 24 A9 14 85
08B0- 25 85 22 A9 18 85 23 20
08B8- 58 FC A9 9E A0 09 20 26
08C0- 0F A9 A0 20 1B FD C9 C3
08C8- D0 05 A2 00 4C FD 08 C9
08D0- CC D0 05 A2 02 4C FD 08
08D8- C9 C5 D0 05 A2 04 4C FD
08E0- 08 C9 CD D0 05 A2 06 4C
08E8- FD 08 C9 D2 D0 05 A2 08
08F0- 4C FD 08 C9 D4 F0 1C 20
08F8- DF 10 4C C1 08 20 FD FB

0900- 20 62 FC 20 62 FC BD 36
0908- 08 85 EB BD 37 08 85 EC
0910- 6C EB 00 A5 00 C9 FF F0
0918- 42 20 C6 10 20 58 FC 20
0920- D3 10 20 62 FC 20 62 FC
0928- A9 34 A0 09 20 26 0F C6
0930- 00 4C A5 08 26 45 52 54
0938- 53 49 47 45 52 4E 45 60
0940- 53 41 50 60 5A 45 56 41
0948- 67 4E 60 53 55 4F 56 60
0950- 7A 60 4E 4F 49 54 4E 45
0958- 54 54 41 A2 00 BD 00 78
0960- 95 00 CA D0 F8 20 58 FC
0968- A9 8D A0 09 20 26 0F 20
0970- D3 10 20 62 FC 4C D0 03
0978- 14 A0 BA A0 D2 D5 C5 D4
0980- C3 C5 CC A0 C5 C4 A0 CF
0988- D2 C5 CD D5 CE 10 D2 C5

séance de travail, éventuellement en changeant de lecteur, sinon tapez T pour Terminer.

Il existe un contrôle, lors de la demande de sortie du programme, afin de ne pas oublier d'enregistrer la séance de travail. Une seconde sortie, immédiatement après la première, vous permet de quitter le programme sans enregistrement.

Contrôles effectués par le programme

- Tout RETURN à vide renvoie au menu principal.
- Les numéros des titres ne peuvent être nuls ou supérieurs au maximum autorisé par le catalogue (en tout état de cause <= 255).
- Par conséquent, il est impossible de demander le transfert d'une zone vers une autre franchissant la limite du catalogue, par exemple 1,50, 80 (pour un catalogue normal contenant au plus 105 titres), qui déplace les titres 1 à 50 vers 80 à 129.
- Le nombre de réponses doit correspondre à ce qui est attendu.
- En cas de non respect des règles ci-dessus, le programme vous repose la question.
- On ne peut pas effacer le titre d'un fichier présent.
- Rappel à l'ordre en cas de sortie du programme sans

enregistrement.

Une disquette dont le catalogue est modifié par cet utilitaire ne peut plus être copiée entièrement par le programme FID, les commentaires provoquant une I/O ERROR. Il faut donc copier les fichiers en sautant les commentaires. Le programme COPYA fonctionne parfaitement.

Technique

L'utilitaire est entièrement écrit en assembleur (Lisa 1.5), d'une part nécessaire pour la rapidité d'exécution, d'autre part pour le plaisir...

Globalement, le catalogue est chargé en MEV, manipulé et enfin réécrit sur la disquette. Le programme ne touche pas à la VTOC (piste \$11, secteur 0).

- Possibilité de lire n'importe quel catalogue commençant en piste \$11, secteur \$F, en utilisant les pointeurs de chaînage d'un secteur avec le suivant, autorisant un maximum de 255 titres. En récupérant les pistes 0 à 2, il est possible de stocker environ 260 fichiers de 2 secteurs chacun, un secteur liste piste/secteur et un secteur données, sur une seule disquette.
- Assemblage en \$801 pour éliminer le SYNTAX ERROR provoqué par NEW, LIST ou RUN après retour au Basic

(\$800 doit contenir 0).

- Sauvegarde de la page zéro en \$7800 et débranchement du DOS à l'initialisation de la routine. La mise hors service du DOS s'explique par la nécessité de pouvoir manipuler des mots réservés du DOS, sans obtenir le fatal NOT DIRECT COMMAND.
- En MEV, le catalogue est stocké à partir de l'adresse \$2000. Une zone de transition pour les déplacements commence en \$4600.
- L'entrée des commentaires entraîne la mise du numéro de piste de la liste piste/secteur à une valeur supérieure à \$22 et inférieure à \$FF (\$30) ; en d'autres termes, le commentaire apparaît pour le DOS ni comme un fichier effacé (il ne s'afficherait pas lors d'un CATALOG), ni comme un fichier présent (une tentative de chargement entraînerait une I/O ERROR). C'est ce numéro insolite que FID ne digère pas.
- La routine d'écriture sur disquette est modifiée de façon à pouvoir enregistrer un catalogue plus long que la normale. Elle commence par le secteur \$F, piste \$11 puis utilise les octets de chaînage pour les secteurs suivants.
- Retour au Basic en rebranchant le DOS, et en restituant la page zéro.



0990- C3 CE C1 CC C5 D2 A0 D2
0998- D5 CF D0 A0 A6 C0 5F A0
09A0- BA A0 D8 C9 CF C8 C3 A0
09A8- C5 D2 D4 CF D6 C0 A0 C5
09B0- CE C9 CD D2 C5 DD D4 DB
09B8- A0 A0 A0 C5 D2 D5 D4 C9
09C0- D2 C3 C5 C5 DD D2 DB A0
09C8- A0 A0 A0 A0 D2 C5 C9 C6
09D0- C9 C4 CF DD CD DB A0 CE
09D8- CF C9 D4 C9 C4 DD C5 DB
09E0- A0 A0 A0 A0 A0 A0 C5 D2
09E8- D5 D4 C3 C5 DD CC DB A0
09F0- A0 A0 D4 CE C5 CD C5 C7
09F8- D2 C1 C8 DD C3 DB A9 0C
0A00- A0 0B 20 26 0F A2 03 A9
0A08- FD A0 09 20 43 0F A9 00
0A10- 85 01 A5 19 18 65 1A 38
0A18- E9 02 85 1B A5 18 C5 1B

0A20- B0 3D A5 25 C9 14 B0 15
0A28- A5 25 85 08 A9 00 85 24
0A30- A9 14 85 25 85 22 A9 18
0A38- 85 23 20 58 FC 20 D3 10
0A40- A9 3D A0 0B 20 26 0F A9
0A48- 00 A6 18 E8 20 24 ED A9
0A50- 48 A0 0B 20 26 0F 20 62
0A58- FC 20 62 FC 4C FE 09 A5
0A60- 01 D0 05 20 C6 10 C6 01
0A68- 20 62 FC C6 19 A9 00 85
0A70- 1C 85 75 A5 19 0A 0A 0A
0A78- 38 E5 19 18 69 01 18 65
0A80- 1C 85 1D AA A9 00 20 24
0A88- ED A2 02 20 4A F9 A9 04
0A90- 85 24 A9 20 18 65 19 85
0A98- CF A5 1C 0A 0A 0A 0A 0A
0AA0- 18 65 1C 65 1C 65 1C 69
0AA8- 0B 85 CE A2 00 A1 CE F0

0AB0- 10 C9 FF F0 0C C9 30 D0
0AB8- 04 A9 A6 D0 06 A9 AB D0
0AC0- 02 A9 AD 20 FD FB E6 24
0AC8- A2 1E A0 03 B1 CE 20 FD
0AD0- FB E6 CE CA 8A D0 F3 A5
0AD8- 75 F0 01 60 A9 8D 20 7C
0AE0- FB E6 1C A5 1C C9 07 D0
0AE8- 8A E6 19 A5 1B C5 19 90
0AF0- 03 4C 6D 0A 20 62 FC A9
0AF8- A0 20 1B FD C9 A0 D0 09
0B00- E6 19 A9 01 85 1A 4C 12
0B08- 0A 4C A5 08 30 BA A0 C5
0B10- D2 C9 CC A0 C1 A0 D3 D2
0B18- D5 C5 D4 C3 C5 D3 A0 C5
0B20- C4 A0 C5 D2 C2 CD CF CE
0B28- A0 D4 C5 A0 D2 D5 C5 D4
0B30- C3 C5 D3 A0 F2 E5 B1 A0
0B38- D5 C4 A0 EF CE 0A A0 D4

0B40-	CE C5 CD C5 CC D5 C5 D3	0D28-	0F 20 62 FC A9 FF 85 75	0F10-	C5 CE C9 CD D2 C5 D4 A0
0B48-	09 D3 D2 D5 C5 D4 C3 C5	0D30-	A5 19 85 1D A5 FB 85 19	0F18-	D4 CE C5 CD C5 D2 D4 D3
0B50-	D3 A0 20 62 FC A9 06 85	0D38-	A5 1D 20 83 0A A9 06 85	0F20-	C9 C7 C5 D2 CE C5 85 EB
0B58-	24 A9 51 A0 0C 20 26 0F	0D40-	24 A9 80 20 6C FD 8A D0	0F28-	84 EC A0 00 B1 EB AA A8
0B60-	A9 A0 20 1B FD 48 20 FD	0D48-	63 A9 40 A0 0E 20 26 0F	0F30-	B1 EB C9 C0 F0 08 20 FD
0B68-	FB 68 C9 C4 F0 0A C9 C5	0D50-	A9 A0 20 1B FD C9 CF F0	0F38-	FB CA 8A D0 F2 60 20 62
0B70-	F0 5C 20 DF 10 4C 52 0B	0D58-	19 C9 EF F0 15 A9 CE 20	0F40-	FC D0 F6 85 EB 84 EC 86
0B78-	20 62 FC 20 62 FC A9 92	0D60-	FD FB A9 CF 20 FD FB A9	0F48-	CE A9 A0 20 6C FD E0 00
0B80-	A0 0C 20 26 0F A2 05 A9	0D68-	CE 20 FD FB 20 62 FC 4C	0F50-	D0 03 4C B7 08 E4 CE 90
0B88-	7D A0 0B 20 43 0F A5 1A	0D70-	EE 0D A9 CF 20 FD FB A9	0F58-	18 C6 CE 46 CE A5 CE 85
0B90-	C5 19 B0 07 AA A5 19 85	0D78-	D5 20 FD FB A9 C9 20 FD	0F60-	75 BD FF 01 C9 AC D0 02
0B98-	1A 86 19 38 E5 19 85 FC	0D80-	FB 20 62 FC A0 00 B1 F9	0F68-	C6 CE CA D0 F4 A5 CE F0
0BA0-	A5 1B 18 65 FC C5 06 F0	0D88-	F0 04 C9 23 90 08 A9 FF	0F70-	0C 68 68 A5 EC 48 A5 EB
0BA8-	08 90 06 20 D3 10 4C 7E	0D90-	91 F9 A0 03 D0 4D 20 62	0F78-	48 20 DF 10 60 A0 00 A2
0BB0-	0B 20 C6 10 A5 19 C5 1B	0D98-	FC 20 62 FC 20 D3 10 A9	0F80-	01 A5 75 F0 70 A9 AC 85
0BB8-	B0 10 A5 1A 85 19 A5 1B	0DA0-	65 A0 0E 20 26 0F 20 62	0F88-	FF BD 00 02 C5 FF D0 09
0BC0-	18 65 FC 85 1A E6 FC 4C	0DA8-	FC 4C A5 08 E0 21 90 02	0F90-	CA BD 00 02 29 0F 4C D7
0BC8-	EB 0B A5 1B D0 F5 20 62	0DB0-	A2 20 86 19 A0 00 B1 3C	0F98-	0F E8 BD 00 02 C5 FF F0
0BD0-	FC 20 62 FC A9 E0 A0 0C	0DB8-	F0 0A C9 30 F0 06 C9 FF	0FA0-	1E 20 FB 0F 99 19 00 CA
0BD8-	20 26 0F A2 03 A9 D3 A0	0DC0-	F0 02 D0 10 A9 30 91 3C	0FA8-	CA BD 00 02 C9 B0 F0 32
0BE0-	0B 20 43 0F A9 01 85 FC	0DC8-	98 A0 21 91 3C C8 91 3C	0FB0-	C9 B1 D0 15 18 B9 19 00
0BE8-	20 C6 10 A5 19 20 15 10	0DD0-	A0 02 91 3C A0 03 A2 00	0FB8-	69 64 E8 E8 4C D7 0F CA
0BF0-	A9 00 85 42 A9 46 85 43	0DD8-	BD 00 02 91 3C E8 C8 E4	0FC0-	BD 00 02 20 FB 0F 4C D7
0BF8-	A0 00 20 2C FE A5 F9 85	0DE0-	19 90 F5 A9 A0 C0 21 B0	0FC8-	0F C9 B2 D0 A4 E8 E8 B9
0C00-	42 A5 FA 85 43 A5 1A 20	0DF8-	05 91 3C C8 D0 F7 E6 FB	0FD0-	19 00 18 69 C8 B0 9A F0
0C08-	15 10 A0 00 20 2C FE A9	0DF0-	A5 FB 85 19 A9 01 85 1A	0FD8-	98 C5 06 F0 02 B0 92 99
0C10-	00 85 3C A9 46 85 3D A9	0DF8-	20 62 FC 4C 12 0A 1C BA	0FE0-	19 00 E8 E8 E8 C8 C6 75
0C18-	22 85 3E A9 46 85 3F A5	0E00-	A0 D2 C5 C9 C6 C9 C4 CF	0FE8-	F0 0B A5 75 C9 FF D0 99
0C20-	F9 85 42 A5 FA 85 43 A0	0E08-	CD A0 C1 A0 C5 D2 D4 C9	0FF0-	A9 00 85 00 60 A9 8D 85
0C28-	00 20 2C FE A5 19 C5 1A	0E10-	D4 A0 D5 C4 A0 CF D2 C5	0FF8-	FF D0 8E 29 0F 99 19 00
0C30-	B0 07 C6 19 C6 1A 4C 3D	0E18-	CD D5 CE 24 A1 AE AE AE	1000-	CA BD 00 02 29 0F 9D 00
0C38-	0C E6 19 E6 1A C6 FC D0	0E20-	AE A1 AE AE AE AE A1 AE	1008-	02 0A 0A 18 7D 00 02 0A
0C40-	AA 20 62 FC E6 FB A5 FB	0E28-	AE AE AE A1 AE AE AE AE	1010-	79 19 00 E8 60 38 E9 01
0C48-	85 19 A9 01 85 1A 4C 12	0E30-	A1 AE AE AE AE A1 AE AE	1018-	85 1D A9 00 85 1E A2 08
0C50-	0A 40 A0 BA A0 D8 C9 CF	0E38-	AE AE A0 A0 A0 A0 A0 A0	1020-	A5 1D 85 FB A5 1E 06 FB
0C58-	C8 C3 A0 C5 D2 D4 CF D6	0E40-	24 BA A0 CE AF CF A0 BF	1028-	2A C9 07 90 04 E9 07 E6
0C60-	C0 C0 C5 CE CF DA A0 C5	0E48-	A0 C5 D2 D4 C9 D4 A0 C5	1030-	FB CA D0 F2 85 1C A9 20
0C68-	CF D5 A7 C4 A0 D4 CE C5	0E50-	C3 A0 D2 C5 C3 C1 C6 C6	1038-	18 65 FB 85 3D 85 3F 85
0C70-	CD C5 C3 C1 CC D0 C5 C4	0E58-	C5 A0 D3 D5 CF D6 AD DA	1040-	FA A5 1C 0A 0A 0A 0A 0A
0C78-	A0 AD A0 DD C4 DB A0 A0	0E60-	C5 CC D5 CF D6 21 45 4D	1048-	18 65 1C 65 1C 65 1C 69
0C80-	A0 A0 A0 A0 C0 C5 C7 CE	0E68-	49 52 50 50 55 53 60 4E	1050-	0B 05 3C 05 F9 18 69 22
0C88-	C1 C8 C3 C5 A0 AD A0 DD	0E70-	4F 4E 60 52 45 49 48 43	1058-	85 3E 60 20 D3 10 A9 E8
0C90-	C5 DB 4D BA A0 CF CF C9	0E78-	49 46 60 7A 60 45 4C 42	1060-	A0 0E 20 26 0F AD B3 10
0C98-	D4 C1 CE C9 D4 D3 C5 C4	0E80-	49 53 53 4F 50 4D 49 20	1068-	09 30 20 FD FB A9 F5 A0
0CA0-	A0 C1 CC A0 C5 C4 A0 AE	0E88-	5B 10 20 62 FC A9 02 8D	1070-	0E 20 26 0F A9 A0 20 1B
0CA8-	D0 D5 D3 A0 C5 D4 C9 CD	0E90-	BD 10 A5 18 85 FC A9 11	1078-	FD 48 20 FD FB 68 C9 CF
0CB0-	C9 CC A0 C1 CC A0 D4 C5	0E98-	8D D5 10 A9 0F 9D B6 10	1080-	F0 26 C9 EF F0 22 C9 B1
0CB8-	D2 C5 C3 C1 CC D0 C5 C4	0EA0-	A9 20 8D BA 10 85 EC A9	1088-	D0 06 29 FF 8D B3 10 60
0CC0-	A0 C1 A0 C5 CE CF DA A0	0EA8-	00 8D B9 10 85 EB 20 A9	1090-	C9 B2 D0 06 29 FF 8D B3
0CC8-	C1 CC A0 C5 C4 A0 D3 C5	0EB0-	10 C6 FC A5 FC C9 FF F0	1098-	10 60 A9 00 85 24 C6 25
0CD0-	D4 C9 CD C9 CC A0 D3 C5	0EB8-	15 EE BA 10 A0 01 B1 EB	10A0-	20 9C FC 68 68 4C A5 08
0CD8-	CC A0 DA C5 D2 D4 CE C5	0EC0-	8D B5 10 C8 B1 EB 8D B6	10A8-	60 A9 10 A0 B1 20 D9 03
0CE0-	26 BA A0 D2 C5 C3 C1 CC	0EC8-	10 E6 EC 4C AF 0F 20 C6	10B0-	60 01 60 01 00 11 0F C2
0CE8-	D0 C5 C4 A0 C1 A0 D2 C5	0ED0-	10 20 58 FC 20 62 FC A9	10B8-	10 00 20 00 00 01 00 00
0CF0-	C9 C8 C3 C9 C6 A0 C5 C4	0ED8-	0F A0 0F 20 26 0F 20 62	10C0-	60 01 00 01 EF D8 A9 00
0CF8-	A0 AE EF CE A0 D3 C5 CC	0EE0-	FC A9 FF 85 00 4C A5 08	10C8-	85 22 A9 14 85 23 A5 08
0D00-	A0 DA C5 D2 D4 CE C5 A9	0EE8-	0C 20 2E 0F 0E 20 12 15	10D0-	85 25 60 A0 C0 20 E9 FB
0D08-	FE A0 0D 20 26 0F A2 01	0EF0-	05 14 03 05 0C 19 20 3A	10D8-	A9 20 A0 C0 20 E6 FB A9
0D10-	A9 06 A0 0D 20 43 0F 20	0EF8-	20 29 0E 2F 0F 2F 32 2F	10E0-	20 A0 C0 20 E6 FB 60 A2
0D18-	C6 10 A5 19 20 15 10 20	0F00-	31 28 20 1A 05 0D 12 09	10E8-	00 CA D0 FD 60
0D20-	62 FC A9 1B A0 0E 20 26	0F08-	06 0E 0F 03 20 2D 20 16		

Programme

LECTCAT.SOURCE

(Assembleur LISA 1.5)

```

1      NLS
2 ;=====
3 ;=
4 ;=  EDITEUR DE CATALOGUE DOG 3.3 =
5 ;=
6 ;= ( CATALOGUE COMMENCANT EN =
7 ;=  PISTE $11, SECTEUR $0F ) =
8 ;=
9 ;=====
10 ;
11 FINFLG EPZ $0
12 LECFLG EPZ $1
13 ADR     EPZ $EB          ; JMP (ADR)
14 FLAG   EPZ $CE
15 CH     EPZ $24
16 CV     EPZ $25
17 SECT   EPZ $18          ; NBRE SECT
18 C1     EPZ $19          ; STOCKE
19 C2     EPZ $1A          ; LES
20 NBSECT EPZ $1B          ; DONNEES
21 F      EPZ $1C
22 NUMFIC EPZ $1D
23 NUMMAX EPZ $06
24 CVANT  EPZ $08
25 A1L    EPZ $3C
26 A1H    EPZ $3D
27 A2L    EPZ $3E
28 A2H    EPZ $3F
29 A4L    EPZ $42
30 A4H    EPZ $43
31 FLAG2  EPZ $75
32 DESTL  EPZ $F9
33 DESTH  EPZ $FA
34 C      EPZ $FB
35 BOUCLE EPZ $FC
36 IN     EQU $200
37 RWTS   EQU $3D9
38 AMPERV EQU $3F5
39 SGNFLT EQU $E301
40 LINPTR EQU $ED24
41 ORIG   EQU $2002
42 PRBL2  EQU $F94A
43 CTRL.S EQU $FB7C
44 BELL2  EQU $FBE6
45 STORAV EQU $FBF0
46 VIDOUT EQU $FBFD
47 KEYIN  EQU $FD1B
48 CR     EQU $FC62
49 CLREOL EQU $FC9C
50 GETLIN EQU $FD6C
51 MOVE   EQU $FE2C
52 ;
53      ORG $801
54 ;
55 ;-----
56 ;      INITIALISATION &

```

```

57 ;-----
58 ;
59 BEGIN  LDA #$4C          ; $4C = JUMP
60      STA AMPERV
61      LDA #DEBPGM
62      STA AMPERV+1
63      LDA /DEBPGM
64      STA AMPERV+2
65 ;
66 ;-----
67 ;      SAUVEGARDE PAGE 0
68 ;-----
69 ;
70 DEBPGM LDX #$00          ; 256 BOUCLES
71      STX FINFLG          ; DRAPEAU ZERO
72 PGO    LDA $00,X
73      STA $7800,X
74      DEX
75      BNE PGO
76 ;
77 ; DEBRANCHE LE D.O.S.
78 ;
79      LDA #$F0
80      STA $36
81      LDA #$FD
82      STA $37
83      LDA #$1B
84      STA $38
85      LDA #$FD
86      STA $39
87 ;
88      JSR $FC58
89      LDA #18
90      STA CV
91      JMP MENU
92 ;
93 ;-----
94 ;      TABLE D'ADRESSE
95 ;-----
96 ;
97 TABLE ADR INIT
98      ADR LECTUR
99      ADR EDIT
100     ADR MODIF
101     ADR ECRIT
102     ADR FIN
103 ;
104 ;=====
105 ;=
106 ;=  INITIALISATION LECTURE -
107 ;=
108 ;=====
109 ;
110 INIT  JSR OK?
111      JSR WINDOW
112      JSR CR
113 ;
114      LDA #$00
115      STA FINFLG
116      STA SECT
117      LDA #$11

```



```

118      STA IBTRK
119      LDA #$0F
120      STA IBSECT
121      LDA #$20
122      STA IBBUFP+1
123      LDA #$1          ; LECTURE
124      STA IBCMD
125      LDA #ORIG-1
126      STA $CE
127      LDA /ORIG-$100
128      STA $CF
129 ;
130 ;-----
131 ;      DEBUT DE LECTURE
132 ;-----
133 ;
134 DEBUT JSR DOS
135      INC $CF
136      LDA #$0
137      TAY
138      CMP ($CE),Y      ; No PISTE=0 ?
139      BNE SUITE
140      INY
141      CMP ($CE),Y      ; No SECT=0?
142      BNE SUITE
143      LDA SECT          ; CALCUL
144      CLC
145      ADC #1
146      ASL              ; NUMMAX
147      ASL              ; =(SECT+1)*7
148      ASL              ; (SECT+1)*8
149      SEC
150      SBC SECT
151      STA NUMMAX
152      DEC NUMMAX        ; NUMMAX =
                          ; (SECT+1)*8-SECT-1
153      JMP MENU
154 SUITE LDY #$0
155      LDA ($CE),Y      ; No PISTE
156      STA IBTRK
157      INY
158      LDA ($CE),Y      ; No SECTEUR
159      STA IBSECT
160      INC IBBUFP+1
161      INC SECT
162      JMP DEBUT
163 ;
164 ;-----
165 ;=
166 ;=      MENU GENERAL
167 ;=
168 ;-----
169 ;
170 MENU  LDA CV
171      STA CVANT
172      LDA #0
173      STA CH
174      LDA #20
175      STA CV
176      STA $22
177      LDA #24

178      STA $23
179      JSR $FC58
180 M1    LDA #MESS1
181      LDY /MESS1
182      JSR AFFICH
183 ;
184 ;-----
185 ;      SAISIE DE LA REPOSE
186 ;-----
187 ;
188 REP1  LDA #$A0
189      JSR KEYIN
190      CMP #"C"          ; CHARGEMENT ?
191      BNE L?
192      LDX #0
193      JMP SAUT
194 L?    CMP #"L"          ; LECTURE ?
195      BNE E?
196      LDX #2
197      JMP SAUT
198 E?    CMP #"E"          ; EDITON ?
199      BNE M?
200      LDX #4
201      JMP SAUT
202 M?    CMP #"M"          ; MODIF ?
203      BNE R?
204      LDX #6
205      JMP SAUT
206 R?    CMP #"R"          ; ENREGIST?
207      BNE T?
208      LDX #8
209      JMP SAUT
210 T?    CMP #"T"          ; TERMINER ?
211      BEQ FIN
212      JSR RETL
213      JMP REP1
214 SAUT  JSR VIDOUT
215      JSR CR
216      JSR CR
217 ;
218      LDA TABLE,X
219      STA ADR
220      LDA TABLE+1,X
221      STA ADR+1
222      JMP (ADR)
223 ;
224 ;-----
225 ;      RETROUVE PAGE0 INITIALE
226 ;-----
227 ;
228 FIN   LDA FINFLG
229      CMP #$FF
230      BEQ OKFIN
231      JSR WINDOW
232      JSR $FC58
233      JSR CLOCHE
234      JSR CR
235      JSR CR
236      LDA #RECMSS
237      LDY /RECMSS
238      JSR AFFICH

```


239	DEC FINFLG	296	CMP #20	
240	JMP MENU	297	BCS M3	; FENETRE MENU
241 ;			DEJA POSITIONNEE	
242 RECMSS HEX 26		298	LDA CV	
243 BLK "ERTSIGERNE SAP ZEVA'N SUOV :		299	STA CVANT	
NOITNETTA"		300	LDA #0	
244 ;		301	STA CH	
245 OKFIN LDX #\$00	; 256 BOUCLES	302	LDA #20	
246 PAGO LDA \$7800,X		303	STA CV	
247 STA \$00,X		304	STA \$22	
248 DEX		305	LDA #24	
249 BNE PAGO		306	STA \$23	
250 ;		307	JSR \$FC58	
251 JSR \$FC58		308 M3	JSR CLOCHE	
252 LDA #MESSAG		309	LDA #MESS3	
253 LDY /MESSAG		310	LDY /MESS3	
254 JSR AFFICH		311	JSR AFFICH	
255 JSR CLOCHE		312	LDA #0	
256 JSR CR		313	LDX SECT	
257 ;		314	INX	
258 JMP \$3D0	; RETOUR BASIC	315	JSR LINPTR	;AFFICHE SECT
259 MESS0 STR " : RUETCEL ED OREMUN"		316	LDA #MESS3B	
260 ;		317	LDY /MESS3B	
261 MESSAG STR "RECNALE RUOP &@"		318	JSR AFFICHE	
262 ;		319	JSR CR	
263 MESS1 STR " : XIOHC ERTOV@ ENIMRE]T[320	JSR CR	
ERUTIRCEE]R[REIFIDO]M[NOITID]E[321	JMP LECTUR	
ERUTCE]L[TNEMEGRAH]C["		322 ONLIT	LDA LECFLG	
264 ;		323	BNE ONLIT2	
265 ;=====		324	JSR WINDOW	
266 ;=	=	325	DEC LECFLG	
267 ;= LECTURE DU BUFFER	=	326 ONLIT2	JSR CR	
268 ;=	=	327	DEC C1	;C1=C1-1
269 ;=====		328 LECT2	LDA #00	
270 ;		329	STA F	;F=0
271 LECTUR LDA #MESS2		330	STA FLAG2	;0=LIRE 1 SECT
272 LDY /MESS2		331 NUMERO	LDA C1	
273 JSR AFFICH		332	ASL	
274 REP2 LDX #\$03		333	ASL	
275 LDA #LECTUR-1		334	ASL	
276 LDY /LECTUR		335	SEC	
277 JSR SAISIE		336	SBC C1	; A->C*7
278 ;		337	CLC	
279 ;-----		338	ADC #01	
280 ;		339	CLC	
281 LDA #0		340	ADC F	; A->F+1+7*C
282 STA LECFLG		341	STA NUMFIC	
283 TEST LDA C1		342 PRITIL	TAX	
284 CLC		343	LDA #0	
285 ADC C2		344	JSR LINPTR	
286 SEC		345	LDX #2	
287 SBC #\$02		346	JSR PRBL2	;AFF X=2 SPACE
288 STA NBSECT	; C1+C2-2 ->	347	LDA #\$4	
NOMBRE DE SECTEURS A LIRE		348	STA CH	
289 LDA SECT		349	LDA /ORIG	
290 CMP NBSECT		350	CLC	
291 BCS ONLIT	; I < C1+C2-2	351	ADC C1	
292 ;-----		352	STA SCF	
293 ; DEPASSEMENT DU NOMBRE DE SECTEURS		353	LDA F	
294 ;-----		354	ASL	
295 LDA CV		355	ASL	

356	ASL		416 ;
357	ASL		417 ;=====
358	ASL	; F*32	418 ;=
359	CLC		419 ;= EDITION DU CATALOGUE
360	ADC F		420 ;=
361	ADC F		421 ;=====
362	ADC F	; F*35	422 ;
363	ADC #ORIG+9		423 EDIT JSR CR
364	STA \$CE		424 LDA #\$06
365	LDX #\$00		425 STA CH
366	LDA (\$CE,X)		426 M4 LDA #MESS4
367	BEQ MOINS		427 LDY /MESS4
368	CMP #\$FF		428 JSR AFFICH
369	BEQ MOINS		429 ;
370	CMP #\$30		430 ;-----
371	BNE PRESEN		431 ; SAISIE DU CHOIX
372	LDA #"&"	; COMMENTAIRE	432 ;-----
373	BNE MOINS+2	; JMP	433 ;
374	PRESEN LDA #"+"	; FICH PRESENT	434 LDA #" "
375	BNE MOINS+2		435 JSR KEYIN
376	MOINS LDA #"-"	; FICH EFFACE	436 PHA
377	JSR VIDOUT		437 JSR VIDOUT
378	INC CH		438 PLA
379	LDX #\$1E	; 30 BOUCLES	439 CMP #"D"
380	BCLA LDY #\$03		440 BEQ DEPLAC
381	LDA (\$CE),Y		441 CMP #"E"
382	JSR VIDOUT		442 BEQ ECHNGE
383	INC \$CE		443 JSR BELL
384	DEX		444 JMP EDIT
385	TXA		445 ;
386	BNE BCLA		446 ;=====
387	LDA FLAG2	; 0=LIRE 1 SECT	447 ;=
388	DEQ CONT		448 ;= DEPLACEMENT
389	RTS	; SINON, ->MODIF	449 ;=
390	CONT LDA #\$8D	; C-R	450 ;-----
391	JSR CTRL.S	; PAUSE	451 ;
392	INC F	; F=F+1	452 DEPLAC JSR CR
393	LDA F		453 JSR CR
394	CMP #\$07		454 M5 LDA #MESS5
395	BNE NUMERO		455 LDY /MESS5
396	INC C1	; C=C+1	456 JSR AFFICH
397	LDA NBSECT		457 ;
398	CMP C1		458 ;-----
399	BCC *+5		459 ; SAISIE DE LA REPONSE
400	JMP LECT2		460 ;-----
401	JSR CR		461 ;
402	LDA #\$A0		462 REP5 LDX #\$05
403	JSR KEYIN		463 LDA #M5-1
404	CMP #" "		464 LDY /M5
405	BNE *+11		465 JSR SAISIE
406	INC C1		466 ;
407	LDA #\$01		467 ;-----
408	STA C2		468 ; TESTE SI C2<C1
409	JMP TEST		469 ;-----
410	JMP MENU		470 ;
411 ;			471 LDA C2
412 MESS2 STR ": ERIL A SRUETCES ED ERBMON			472 CMP C1
TE RUETCES rel UD ON"			473 BCS OVER?
413 ;			474 TAX
414 MESS3 STR " TNEMELUES"			475 LDA C1
415 MESS3B STR "SRUETCES "			476 STA C2


```

477      STX C1
478 ;
479 OVER? SEC
480      SBC C1
481      STA BOUCLE
482 ;
483 ;-----
484 ; TESTE SI DEPASSEMENT DE NUMMAX
485 ; PAR LA ZONE DE RECEPTION
486 ;-----
487 ;
488      LDA NBSECT
489      CLC
490      ADC BOUCLE
491      CMP NUMMAX
492      BEQ MONTE?      ; =NUMMAX
493      BCC MONTE?      ; <NUMMAX
494      JSR CLOCHE
495      JMP M5
496 ;
497 MONTE? JSR WINDOW
498      LDA C1
499      CMP NBSECT
500      BCS MONTEE      ; C1>C3
501 ;
502      LDA C2
503      STA C1           ; C1=C2
504      LDA NBSECT
505      CLC
506      ADC BOUCLE      ; C3=C3+BOUCLE
507 MONO   STA C2        ; C2=C3
508      INC BOUCLE
509      JMP BCL5
510 MONTEE LDA NBSECT
511      BNE MONU        ; JMP
512 ;
513 ;=====
514 ;=
515 ;=      ECHANGE      =
516 ;=
517 ;=====
518 ;
519 ECHNGE JSR CR
520      JSR CR
521 M6     LDA #MESS6
522      LDY /MESS6
523      JSR AFFICH
524 ;
525 ;-----
526 ;      SAISIE REPOSE
527 ;-----
528 ;
529 REP6   LDX #S03
530      LDA #MG-1
531      LDY /M6
532      JSR SAISIE
533      LDA #S01      ; 1 BOUCLE
534      STA BOUCLE
535      JSR WINDOW
536 ;
537 ;-----
538 ;
539 BCL5   LDA C1
540      JSR ADDRESS
541      LDA #S00
542      STA A4L
543      LDA #S46
544      STA A4H
545      LDY #S00
546      JSR MOVE
547      LDA DESTL
548      STA A4L
549      LDA DESTH
550      STA A4H
551      LDA C2
552      JSR ADDRESS
553      LDY #S00
554      JSR MOVE
555      LDA #S00
556      STA A1L
557      LDA #S46
558      STA A1H
559      LDA #S22
560      STA A2L
561      LDA #S46
562      STA A2H
563      LDA DESTL
564      STA A4L
565      LDA DESTH
566      STA A4H
567      LDY #S00
568      JSR MOVE
569      LDA C1
570      CMP C2
571      BCS POS
572      DEC C1
573      DEC C2
574      JMP POS+4
575 POS    INC C1
576      INC C2
577      DEC BOUCLE
578      BNE BCL5
579      JSR CR
580      INC C
581      LDA C
582      STA C1
583      LDA #S01
584      STA C2
585      JMP TEST      ; VERS LECTURE
586 MESS4  STR " : XIOHC ERTOVE@ENOZ ENU'D
TNEMECALPED - ]D[      @EGNAHCE - ]E["
587 ;
588 ;
589 MESS5  STR " : NOITANITSED AL ED .PUS
ETIMIL AL TEREALPED A ENOZ AL ED SETIMIL SEL
ZERTNE"
590 ;
591 ;
592 MESS6  STR " : RECALPED A REIHCIF ED .ON
SEL ZERTNE"
593 ;
594 ;=====

```



```

595 ;=
596 ;= MODIFICATION DES TITRES =
597 ;=
598 ;=====
599 ;
600 MODIF LDA #MESS7
601 LDY /MESS7
602 JSR AFFICH
603 ;
604 ;-----
605 ; SAISIE DE LA REPONSE
606 ;-----
607 ;
608 LDX #01
609 LDA #MODIF-1
610 LDY /MODIF
611 JSR SAISIE
612 JSR WINDOW
613 ;
614 ;-----
615 ; CALCUL D'ADRESSE
616 ;-----
617 ;
618 LDA C1
619 JSR ADRESS ; DEBUT EN
DESTL,DESTH
620 JSR CR
621 ;
622 ;-----
623 ; AFFICHAGE DU TITRE A MODIFIER
624 ;-----
625 ;
626 M8 LDA #MESS8
627 LDY /MESS8
628 JSR AFFICH ;NOUVEAU TITRE
629 JSR CR
630 ;
631 LDA #$FF
632 STA FLAG2 ; $FF=1 TITRE
633 LDA C1
634 STA NUMFIC ; No DU TITRE
635 LDA C
636 STA C1 ; No SECTEUR
637 LDA NUMFIC
638 JSR PRITIL
639 ;
640 ;
641 ;-----
642 ; SAISIE DE LA CHAINE
643 ;-----
644 ;
645 LDA #6
646 STA CH
647 LDA #$80 ; CARACT NUL
648 JSR GETLIN
649 ;
650 ; TEST
651 ;
652 TXA
653 BNE BCL7 ; NON VIDE
654 M9 LDA #MESS9

```

```

655 LDY /MESS9
656 JSR AFFICH
657 LDA #" "
658 JSR KEYIN
659 CMP #"O"
660 BEQ OUI? ;PEUT-ON EFF
661 CMP #"o"
662 BEQ OUI? ; IDEM
663 LDA #"N"
664 JSR VIDOUT
665 LDA #"O" ; ON NE VEUT
PAS EFFACER
666 JSR VIDOUT
667 LDA #"N"
668 JSR VIDOUT
669 JSR CR
670 JMP FINI ; VERS LECTURE
671 ;
672 OUI? LDA #"O"
673 JSR VIDOUT
674 LDA #"U"
675 JSR VIDOUT
676 LDA #"I"
677 JSR VIDOUT
678 JSR CR
679 LDY #$00
680 LDA (DESTL),Y
681 BEQ *+6 ; A=0
682 CMP #$23
683 BCC NON ; A < 35
684 LDA #$FF ;FICH SUPPRIME
685 STA (DESTL),Y
686 LDY #$03
687 BNE BCL10-2 ; JMP
688 NON JSR CR
689 JSR CR
690 M10 JSR CLOCHE
691 LDA #MESS10
692 LDY /MESS10
693 JSR AFFICH ; ON NE PEUT
PAS EFFACER
694 JSR CR
695 JMP MENU
696 BCL7 CPX #$21 ;X<33 ?
697 BCC X<33
698 LDX #$20 ;X=32
699 X<33 STX C1 ;SAUVE X
700 ;
701 ;-----
702 ;ON REMPLACE UN TITRE PAR UN COMMENTAIRE?
703 ;-----
704 ;
705 LDY #$00
706 LDA (AIL),Y
707 BEQ COMMEN
708 CMP #$30 ;COMMENTAIRE ?
709 BEQ COMMEN
710 CMP #$FF ;FICH EFFACE ?
711 BEQ COMMEN
712 BNE TITRE
713 COMMEN LDA #$30

```



```

714      STA (A1L),Y      ;      CODE      770 ;=
POUR FICHIER SUPPRIME MAIS APPARAISSANT LORS 771 ;=      ECRITURE SUR DISQUETTE      =
D'UN 'CATALOG'      772 ;=      =
715 ;      773 ;=====
716      TYA      774 ;
717      LDY #S21      ; MISE A      775 ECRIT JSR OK?
718      STA (A1L),Y      ; ZERO DU      776 JSR CR
719      INY      ; NOMBRE DE      777 LDA #S02      ; ECRITURE
720      STA (A1L),Y      ; SECTEURS      778 STA IBCMD
721      LDY #2      779 LDA SECT
722      STA (A1L),Y      ; COD FICH TEXT      780 STA BOUCLE
723 ;      781 LDA #S11
724 ;-----      782 STA IBTRK      ; PISTE $11
725 ;      TRANSFERT IN -> TITRE      783 LDA #S0F
726 ;-----      784 STA IBSECT      ; SECTEUR SF
727 ;      785 LDA #S20
728 TITRE LDY #S03      786 STA IBBUFP+1      ; BUFFER
729 LDM #S00      787 STA ADR+1
730 BCL9 LDA TN,X      788 LDA #0
731 STA (A1L),Y      789 STA IBBUFP      ; EN $2000
732 INX      790 STA ADR
733 INY      791 BCL11 JSR DOS
734 CPX C1      792 DEC BOUCLE
735 BCC BCL9      ; X<C1      793 LDA BOUCLE
736 ;      794 CMP #SFF
737 ;-----      795 BEQ END
738 ;      COMPLETE LE TITRE AVEC      796 INC IBBUFP+1
739 ; DES BLANCS ( MAX. 30 CARACTERES)      797 LDY #1
740 ;-----      798 LDA (ADR),Y      ; No PISTE
741 ;      799 STA IBTRK
742 LDA #" "      800 INY
743 BCL10 CPY #S21      801 LDA (ADR),Y      ; No SECTEUR
744 BCS FINI      ; Y>=30      802 STA IBSECT
745 STA (A1L),Y      803 INC ADR+1
746 INY      804 JMP BCL11
747 BNE BCL10      ; JMP      805 ;
748 FINI INC C      806 ;
749 LDA C      807 ;-----
750 STA C1      808 ;      FIN D'ENREGISTREMENT
751 LDA #S01      809 ;-----
752 STA C2      810 ;
753 JSR CR      811 END JSR WINDOW
754 JMP TEST      ; VERS LECTURE      812 JSR $FC58
755 ;      813 JSR CR
756 ;      814 LDA #MESS13
757 MESS7 STR " : REIFIDOM A ERTIT UD OREMUN"      815 LDY /MESS13
758 ;      816 JSR AFFICH
759 ;      817 JSR CR
760 MESS8 STR"!.....!.....!.....!.....!.....!.....      818 LDA #SFF
"      819 STA FINFLG
761 ;      820 JMP MENU
762 ;      821 ;
763 MESS9 STR " : N/O ? ERTIT EC RECAFFE      822 MESS11 HEX 0C
SUOV-ZELUOV"      823 INV " .ON RUETCEL"
764 ;      824 ;
765 ;      825 MESS12 HEX 19
766 MESS10 HEX 21      ; LONG MESSAGE      826 INV " : )N/O/2/1( ZEMRIFNOC - "
767 BLK "EMIRPPUS NON REIHCIF :      827 ;
ELBISSOPMI"      828 MESS13 STR "ENIMRET TNEMERTSIGERNE"
768 ;      829 ;
769 ;=====      830 ;=====

```


831 ;=	=	891	LDA ADR+1	
832 ;=	SOUS-PROGRAMMES	=	892	PHA ;EMPIL NOUVEL
833 ;=	=		893	LDA ADR
834 ;=====			894	PHA ;ADRESS RET
835 ;			895	JSR BELL
836 ;			896	RTS
837 ;-----			897 ;	
838 ;	AFFICHAGE DU MESSAGE		898 ;-----	
839 ;-----			899 ;	TRAITEMENT DE LA REPONSE
840 ;			900 ;-----	
841 AFFICH STA ADR			901 ;	
842	STY ADR+1		902 TRAITE LDY #\$00	
843	LDY #\$00		903	LDX #\$01
844	LDA (ADR),Y		904	LDA FLAG2 ;PAS DE
845	TAX - ; LONG MESSAGE			VIRGULE A TESTER
846 MESS TAY			905	BEQ T2
847	LDA (ADR),Y		906	LDA #", "
848	CMP #"@" ; CODE POUR		907	STA SFF
	R.C.		908 BCL6	LDA IN,X
849	BEQ R.C.		909	CMP SFF ; VIRGUL OU CR
850 VIDEO JSR VIDOUT			910	BNE T1
851	DEX ; REPONSE		911	DEX ; REPONSE
852	TXA ; SUR 1 OCTET		912	LDA IN,X ; SUR 1 OCTET
853	BNE MESS ; A=A-\$B0		913	AND #\$0F ; A=A-\$B0
854	RTS		914	JMP STORE
855 R.C. JSR CR			915 T1	INX
856	BNE VIDEO+3 ;JMP		916	LDA IN,X
857 ;			917	CMP SFF ; VIRGUL OU CR
858 ;-----			918	BEQ DEUX ; OUI : 2
859 ;	SAISIE DES REPONSES			OCTETS A TRAITER
860 ;-----			919	JSR OCT2 ; NON : 3
861 ;			920	STA C1,Y
862 SAISIE STA ADR			921	DEX
863	STY ADR+1		922	DEX
864	STX FLAG ; NB MIN OCTET		923	LDA IN,X
865	LDA #" " ; SAISIE		924	CMP #\$B0 ; =0 ?
866	JSR GETLIN ; SAISIE		925	BEQ NXT ; OUI -> SUITE
867	CPX #\$00 ; SAISIE		926	CMP #\$B1 ; =1 ?
868	BNE SAISI2 ;VERS MENU		927	BNE DEUX?
869	JMP M1-3 ;SANS MODIF		928	CLC
870 SAISI2 CPX FLAG			929	LDA C1,Y
871	BCC ERREUR		930	ADC #\$64 ; AJOUTE 100
872	DEC FLAG		931	INX
873	LSR FLAG ; NR VIRGULES		932	INX
874	LDA FLAG		933	JMP STORE
875	STA FLAG2		934 DEUX	DEX
876 BCL1 LDA IN-1,X			935	LDA IN,X
877	CMP #", " ; VIRGULE ?		936	JSR OCT2
878	BNE *+4		937	JMP STORE
879	DEC FLAG		938 DEUX?	CMP #\$B2 ; =2 ?
880	DEX ;DEPASSEMENT		939	BNE ERREUR ;DEPASSEMENT
881	BNE BCL1		940	INX
882	LDA FLAG		941	INX
883	BEQ TRAITE		942	LDA C1,Y
884 ;			943	CLC
885 ;-----			944	ADC #\$C8 ; AJOUTE 200
886 ;	TRAITEMENT DES ERREURS		945	BCS ERREUR ; > 255
887 ;-----			946 STORE	BEQ ERREUR ;ENTREE = 0
888 ;			947	CMP NUMMAX
889 ERREUR PLA ;DEPIL ANCIENN			948	BEQ STORE2
890	PLA ;ADRESS RETOUR		949	BCS ERREUR ;ENTREE>NUMMAX

950 STOREZ STA C1,Y		979 ADC C1,Y	
951 NXT INX		980 INX	
952 INX		981 RTS	
953 INX		982 ;	
954 TNY		983 ;-----	
955 DEC FLAG2	; SI 0, PLUS	984 ; CALCULS D'ADRESSES	
	DE VIRGULE A	985 ;-----	
	TESTER	986 ;	
956 BEQ T2		987 ADRESS SEC	
957 LDA FLAG2		988 SBC #\$01	
958 CMP #\$FF		989 STA NUMFIC	; N-1
959 BNE BCL6		990 LDA #\$00	
960 LDA #0		991 STA NUMFIC+1	
961 STA FINFLG	; DRAPEAU ZERO	992 LDX #\$08	
962 RTS		993 LDA NUMFIC	
963 T2 LDA #\$8D	; ON TESTE CR	994 STA C	
964 STA \$FF		995 LDA NUMFIC+1	
965 BNE BCL6	; JMP	996 DIVID ASL C	; DIVISION
966 ;		997 ROL	; PAR 7
967 ;		998 CMP #\$07	; DE
968 OCT2 AND #\$0F	; A: A-\$B0	999 BCC NEXT	; NUMFIC
969 STA C1,Y		1000 SBC #\$07	
970 DEX		1001 INC C	; C->No SECT
971 LDA IN,X		1002 NEXT DEX	

Editeur Plein Ecran

Le Pacha

EPE

Apple][+, //e, //c

- *Listez vos programmes Basic en avant et en arriere.*
- *Modifiez, inserez, effacez des caracteres en plein ecran sans relire les lignes.*
- *Recherchez toute chaîne de caracteres.*
- *Choisissez vous-même les codes de contrôle d'EPE.*
- *Modifiez EPE : le fichier source est sur la disquette.*

150,00 F TTC franco
(bon de commande page 74)

972 AND #\$0F	; A: A-\$B0	1003 BNE DIVID	
973 STA IN,X		1004 STA F	; F->No TITRE
974 ASL		1005 LDA /ORIG	
975 ASL	; A*4	1006 CLC	
976 CLC	; C=0	1007 ADC C	
977 ADC IN,X	; A*4+A	1008 STA A1H	
978 ASL	; A*10	1009 STA A2H	

1010	STA DESTH		RETOUR PUISQUE PAS DE RTS
1011	LDA F		1071 JMP MENU
1012	ASL		1072 OK RTS
1013	ASL		1073 ;
1014	ASL		1074 ;=====
1015	ASL		1075 ;= =
1016	ASL		1076 ;= TABLE IOB ET CARACTERISTIQUES =
1017	CLC		1077 ;= =
1018	ADC F		1078 ;=====
1019	ADC F		1079 ;
1020	ADC F		1080 DOS LDA /IOB
1021	ADC #ORIG+9		1081 LDY #IOB
1022	STA A1L		1082 JSR RWTS
1023	STA DESTL		1083 RTS
1024	CLC		1084 ;
1025	ADC #\$22		1085 IOB HEX 0160
1026	STA A2L		1086 IBDRVN HEX 01
1027	RTS		1087 HEX 00
1028 ;			1088 IBTRK HEX 11
1029 ;-----			1089 IBSECT HEX 0F
1030 ; CONFIRMATION CHOIX			1090 ADR CARACT
1031 ;-----			1091 IBBUFP HEX 0020
1032 ;			1092 HEX 0000
1033 OK? JSR CLOCHE			1093 IBCMD HEX 01
1034 LDA #MESS11			1094 HEX 00006001
1035 LDY /MESS11			1095 ;
1036 JSR AFFICH			1096 CARACT HEX 0001EFD8
1037 ;			1097 ;
1038 LDA IBDRVN			1098 ;-----
1039 ORA #\$30 ; A=A+\$30			1099 ; POSITIONNE LA FENETRE TEXTE
1040 JSR VIDOUT			1100 ;-----
1041 ;			1101 ;
1042 LDA #MESS12			1102 WINDOW LDA #0
1043 LDY /MESS12			1103 STA \$22
1044 JSR AFFICH			1104 LDA #\$14
1045 ;			1105 STA \$23
1046 LDA #" "			1106 LDA CVANT
1047 JSR KEYIN			1107 STA CV
1048 PHA			1108 RTS
1049 JSR VIDOUT			1109 ;
1050 PLA			1110 ;-----
1051 CMP #"O"			1111 ; SON DE CLOCHE
1052 BEQ OK			1112 ;-----
1053 CMP #"o"			1113 CLOCHE LDY #\$C0
1054 BEQ OK			1114 JSR BELL2+3
1055 CMP #"1"			1115 LDA #\$20
1056 BNE LECT2?			1116 LDY #\$C0
1057 AND #\$FF			1117 JSR BELL2
1058 STA IBDRVN			1118 BELL LDA #\$20
1059 RTS			1119 LDY #\$C0
1060 LECT2? CMP #"2"			1120 JSR BELL2
1061 BNE *+8			1121 RTS
1062 AND #\$FF			1122 ;
1063 STA IBDRVN			1123 TEMPO LDX #0
1064 RTS			1124 BCLTEM DEX
1065 LDA #\$00			1125 BNE BCLTEM
1066 STA CH			1126 RTS
1067 DEC CV			1127 ;
1068 JSR CLREOL			1128 LENGTH EQU *-BEGIN
1069 PLA ;DEPILE 2 OCT			1129 PAU
1070 PLA ;DE ADRESSE DE			1130 END



Deux utilitaires et un programme de gestion au sommaire de la Pom's 20 :

- *Screen Maker*, l'équivalent en assembleur du programme de personnalisation des disquettes Mac paru dans la revue n° 16 : Transformer un document MacPaint en une image de 'Boot'.
- Des messages en boîtes, un utilitaire dont le source est abondamment commenté, pour la gestion de boîtes à dialogue.
- Un programme de gestion de comptes bancaires en Basic Microsoft doté des facilités propres au Mac.

80,00 F TTC franco, Bon de commande page 74

Disquette Macintosh
14/15/16

- Programmes Basic des numéros 14, 15 et 16 de Pom's : 'Editeur', 'Mac-Apple II', 'Paint-Basic', et autres programmes "exemples" pour l'illustration des articles de ces numéros.
- Polices 'Los Angeles', 'Cairo', 'Mos Esley', 'Hollywood', 'Manhattan',
- programme 'Localizer',
- programme 'Copie de disque'.

150,00 F TTC franco

Disquette Macintosh
numéro 19

- Nouvelle version de MacWrite (4.5), version "Disk based" qui permet de manipuler de gros fichiers même sur un Macintosh 128 Ko,
- MacPaint 1.5 qui ne perd plus la grille lors des déplacements,
- Lucy, et 5, accessoires de bureau, aux sources largement commentés,
- Programmes exemples liés aux articles.

80,00 F TTC franco

Disquette Macintosh
numéro 18

- Version 4.1 du Finder, plus rapide que la précédente,
- 'System Update' pour implanter les nouveaux Finder et System sur vos disquettes,
- 'BR Demo' pour l'affichage temporisé d'écrans,
- 'Font/DA Mover' pour manipuler polices et accessoires,
- Polices 'Times', 'Courier', 'Symbol' et 'Helvetica',
- Programmes de la revue.

80,00 F TTC franco

Disquette Macintosh
numéro 17

- 'Catalogue sur Imprimante' pour obtenir toutes les informations sur vos fichiers : type, créateur, index, position et longueur des "ressources", icône visible ou non, fichier protégé ou non...
- Routines 'Bload' et 'Bsave' similaires à celles de l'Apple II (pour une sauvegarde et un chargement rapide des fichiers binaires).

80,00 F TTC franco

Puisque nous ne vous aviez besoin, nous

Apple et le logo Apple sont des marques déposées d'Apple Computer, Inc.



Un clavier 63 touches type AZERTY comprenant une accentuation complète et des caractères majuscules/minuscules intégrés.

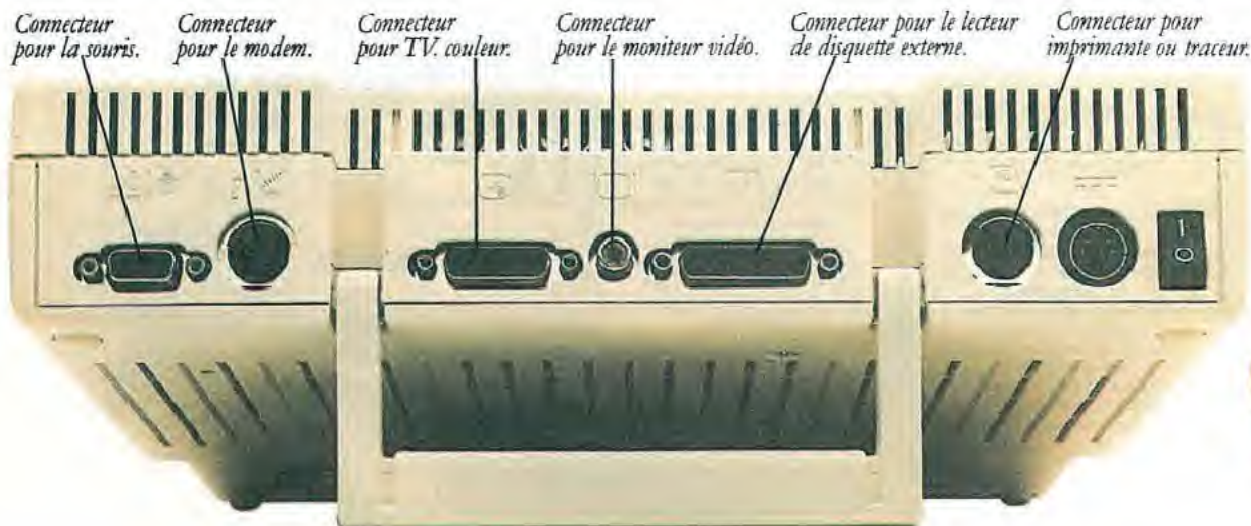


Version Calc



MacPaint

Une des plus grandes bibliothèques de logiciels : programmes compatibles avec l'Apple IIe : jeux, gestion de base de données, analyse financière.



Voici comment l'écran de haute résolution se contrôle très facilement. Si nous vous en avons besoin, en un seul appareil, c'était indispensable.

savons pas de quoi vous avons tout donné.



Budget Familial



Apple Writer



Sorcellerie



Apple Works



Omnis II



Les Oursins malins

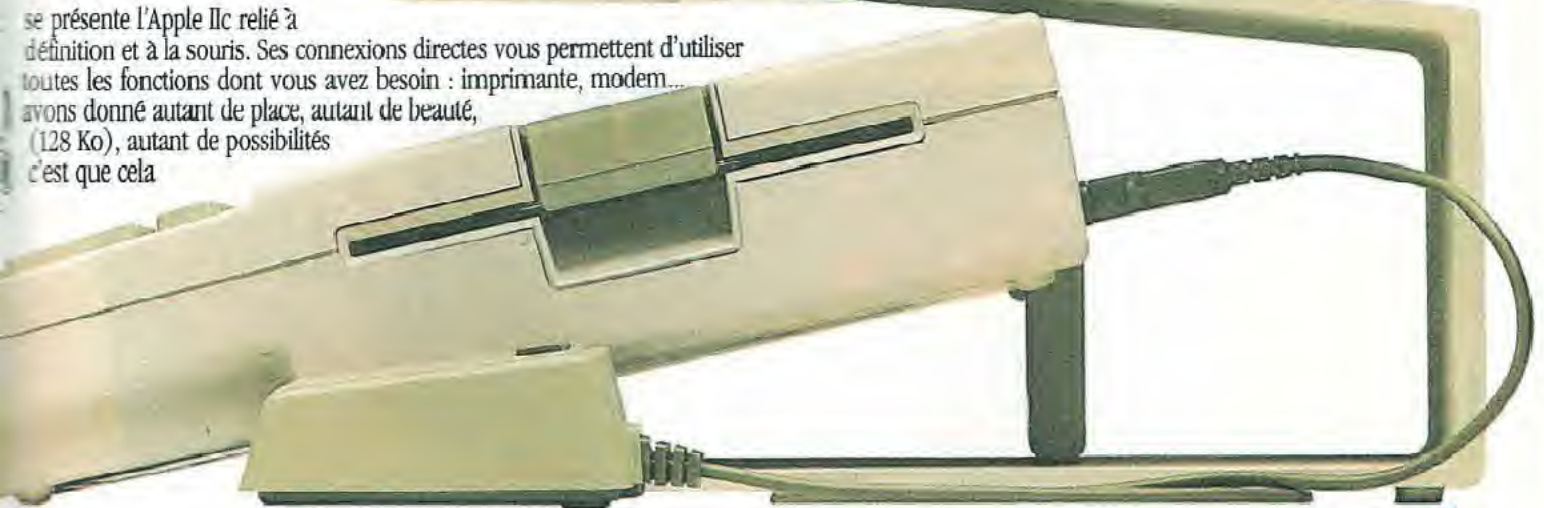
au monde ; 16.000
traitement de texte,
planification.

Voici 8 exemples de ce que vous pouvez afficher sur votre
écran. 8 parmi 16.000 ! à vous de découvrir les autres.

Lorsque vous aurez essayé l'Apple IIc vous ne regretterez pas
notre générosité.



se présente l'Apple IIc relié à
définition et à la souris. Ses connexions directes vous permettent d'utiliser
toutes les fonctions dont vous avez besoin : imprimante, modem...
avons donné autant de place, autant de beauté,
(128 Ko), autant de possibilités
c'est que cela



Apple présente l'Apple IIc.



l'actualité
les bancs d'essai
les guides d'achat
le dossier
les programmes

L'ORDINATEUR L'INDIVIDUEL



LA RÉFÉRENCE EN MICRO-INFORMATIQUE



A L'ORDINATEUR INDIVIDUEL, les rédacteurs, les conseillers techniques, les correspondants à l'étranger, l'équipe entière se mobilise pour vous fournir tous les mois une information complète et de qualité. Le monde de la micro bouge : L'O.I. teste pour vous les micros et logiciels qui apparaissent sur le marché. Il vous dit lesquels choisir et pourquoi. Vous êtes déjà équipé et vous souhaitez tirer le maximum de votre machine ? Les spécialistes de L'O.I. vous livrent conseils, programmes inédits et astuces d'utilisation. Lisez chaque mois L'ORDINATEUR INDIVIDUEL.



Gérard Michel

Des messages en boîtes

Cette petite routine vous permettra d'afficher facilement les messages de vos programmes Basic sous la forme de ces "boîtes à dialogue" dont le Macintosh raffole.

Le principe d'utilisation en est simple, puisque les seules données à fournir sont :

- les dimensions de la boîte, via l'indication du point supérieur gauche et du point inférieur droit du rectangle dans lequel elle s'inscrira (coordonnées de ces deux points par rapport au coin supérieur gauche de la fenêtre d'affichage du Basic) ;
- les dimensions du rectangle dans lequel apparaîtra le texte du message sur une ligne (coordonnées des deux points significatifs calculées cette fois par rapport au coin supérieur gauche de la boîte elle-même) ;
- les dimensions du "pseudo-bouton" de contrôle (coordonnées toujours indiquées par rapport à la boîte). Ce bouton contiendra le message standard "Return S.V.P. " ;
- l'adresse de la variable chaîne de caractères où est stocké le texte du message.

Mise en œuvre

Le code de la routine en langage-machine est implanté dans un tableau de variables entières (CO%(274) dans notre programme d'exemple).

En fait, les premiers éléments de ce tableau contiennent les paramètres de la boîte, du rectangle de texte et du bouton-contrôle :

- CO%(0) à CO%(3) : coordonnées des deux points définissant les dimensions de la boîte ;
- CO%(4) à CO%(7) : idem pour le rectangle de texte ;
- CO%(8) à CO%(11) : idem pour les dimensions du "bouton" ;
- CO%(12) à CO%(19) : définition du texte contenu dans le "bouton".

La zone CO%(20) à CO%(190) sert quant à elle de tampon pour les routines système qui gèrent les fenêtres de dialogue et les événements qui s'y produisent.

Le code de la routine commence donc en CO%(191).

Appel de la routine

Il résulte des instructions :

- A! VARPTR(V\$) avec la nouvelle version du Basic Microsoft

- ou CALL A! (VARPTR(V\$)) avec l'ancienne version où :

- A! = VARPTR(CO%(191)) = adresse de début du code de la routine ;
- V\$ = chaîne de caractères contenant le message à afficher (au sujet du mode de stockage des variables par le Basic et des descripteurs de chaînes, reportez-vous aux cahiers Mac des précédents numéros de Pom's).

La routine dessine la boîte, y place le texte du message, dessine le bouton-contrôle et attend un événement, le seul qui puisse amener la disparition de la boîte et le retour au Basic étant la frappe de la touche "RETURN".


Vous pouvez bien sûr modifier les variables utilisées et leur contenu, ainsi que les positions et dimensions des différents éléments du "dialogue", avant chaque appel, afin de les adapter aux besoins.

Les commentaires qui accompagnent la liste du programme source en assembleur vous permettront de suivre le déroulement des opérations, et notamment le nom et le rôle des différentes routines système utilisées.



Source de la routine

0000	.Trap	_FrontWindow	SA924	Routines utilisées par
0000	.Trap	_NewDialog	\$A97D	le programme.
0000	.Trap	_SetPort	\$A873	Sur un 512 Ko, la défi-
0000	.Trap	_NewControl	\$A954	nition des 'Traps' peut
0000	.Trap	_TextBox	\$A9CE	être remplacée par :
0000	.Trap	_ModalDialog	\$A991	INCLUDE MacTraps.D
0000	.Trap	_CloseDialog	\$A982	
0000	.Trap	_SelectWindow	\$A91F	



```

0000 0064 0064 00DE 0190 Rectangle DC      100,100,190,400      ; Boîte
0008 000F 0014 0024 012C T2      DC      15,20,36,300      ; Rectangle pour texte
0010 003C 0082 0050 0122 T0      DC      60,130,80,290      ; Rectangle pour contrôle
0018 0E 52 65 74 75 72 6E 20 53 2E 56 2E 50 2E 20 00
                                Return DC,B    14,'Return S.V.P. ',0 ; contenu du contrôle
0028 xxxx xxxx xxxx Dstorage DCB    $AA,0      ; Tampon pour boîte à dialogue
017C 0000 ItemHit DC      0      ; Stockage événement dialogue
017E Texte EQU      D4
017E Long EQU      D3
017E
017E 4E56 0000 LINK      A6,#0
0182 226E 0008 MOVEA.L 8(A6),A1      ; adresse descripteur de chaîne
0186 1629 0001 MOVE.B 1(A1),Long      ; longueur de la chaîne
018A 0283 0000 00FF ANDI.L #S0000000FF,Long      ; convertie en un long-mot
0190 2829 0001 MOVE.L 1(A1),Texte
0194 0284 00FF FFFF ANDI.L #S00FFFFFF,Texte      ; adresse de la chaîne dans D4
019A Dialogue
019A 42A7 CLR.L -(SP)
019C A924 _FrontWindow      ; pointeur sur fenêtre active
019E 265F MOVE.L (SP)+,A3      ; qui est stocké dans A3
01A0 42A7 CLR.L -(SP)
01A2 487A FE84 PEA Dstorage
01A6 487A FE58 PEA Rectangle
01AA 42A7 CLR.L -(SP)      ; pas de titre
01AC 1F3C FFFF MOVE.B #-1,-(SP)      ; -1 = $FF = VRAI => Visible
01B0 3F3C 0001 MOVE #1,-(SP)      ; type fenêtre=boîte à dialogue
01B4 2F3C FFFF FFFF MOVE.L #-1,-(SP)      ; la fenêtre vient sur le dessus
01BA 1F3C 0000 MOVE.B #0,-(SP)      ; 0 = FAUX = pas de "GoAway"
01BE 2F3C 0000 0000 MOVE.L #0,-(SP)      ; les autres param. sont nuls
01C4 2F3C 0000 0000 MOVE.L #0,-(SP)
01CA A97D _NewDialog      ; ouvre la boîte à dialogue
01CC 2E17 MOVE.L (SP),D7      ; met son pointeur dans D7
01CE A873 _SetPort      ; boîte = port d'entrée/sortie
01D0 42A7 CLR.L -(SP)
01D2 2F07 MOVE.L D7,-(SP)      ; précise utilisation boîte
01D4 487A FE3A PEA T0
01D8 487A FE3E PEA Return
01DC 1F3C FFFF MOVE.B #-1,-(SP)      ; contrôle visible
01E0 2F3C 0000 0000 MOVE.L #0,-(SP)      ; autres param. à 0, dont type
01E6 2F3C 0000 0000 MOVE.L #0,-(SP)      ; contrôle (0=Return pour nous)
01EC 2F3C 0000 0000 MOVE.L #0,-(SP)
01F2 A954 _NewControl      ; ouvre le "bouton" contrôle
01F4 201F MOVE.L (SP)+,D0      ; enlève son pointeur de la pile
01F6
01F6 2F04 MOVE.L Texte,-(SP)
01F8 2F03 MOVE.L Long,-(SP)
01FA 487A FE0C PEA T2
01FE 3F3C 0001 MOVE #1,-(SP)      ; justification = centré
0202 A9CE _TextBox      ; "dessine" texte dans rect. T2
0204 Attente
0204 42A7 CLR.L -(SP)
0206 487A FF74 PEA ItemHit
020A A991 ModalDialog      ; attend un événement dans boîte
020C 303A FF6E MOVE ItemHit,D0
0210 0C40 0001 CMPI #1,D0      ; si code événement = 1=> Return
0214 66 EE BNE Attente      ; sinon, attendre encore
0216 2F07 MOVE.L D7,-(SP)
0218 A982 _CloseDialog      ; referme la boîte
021A 2F0B MOVE.L A3,-(SP)
021C A91F _SelectWindow      ; fenêtre initiale en activité
021E 2F0B MOVE.L A3,-(SP)
0220 A873 _SetPort      ; et port d'entrée/sortie
0222 4E5E UNLK A6      ; retour au Basic
0224 4E75 RTS

```


Programme de démonstration

DEFINT A-Z

DIM CO(274)

' Les deux premières lignes donnent les paramètres
' de la boîte, du rectangle de texte et du rectangle de
' contrôle, ainsi que le contenu de ce dernier

DATA 100,100,190,400,15,20,36,300,60,130,80

DATA 290,&H0E52,&H6574,&H7572,&H6E20

DATA &H532E,&H562E,&H502E,&H2000

' Stockage de "l'événement dans la boîte"

DATA 0

' Code de la routine

DATA &H4E56,0,&H226E,8,&H1629,1,&H0283,0

DATA &HFF,&H2829,1,&H0284,&HFF,&HFFFF

DATA &H42A7,&HA924,&H265F,&H42A7,&H487A

DATA &HFE84,&H487A,&HFE58,&H42A7,&H1F3C

DATA &HFFFF,&H3F3C,1,&H2F3C,&HFFFF

DATA &HFFFF,&H1F3C,0,&H2F3C,0,0,&H2F3C,0

DATA 0,&HA97D,&H2E17,&HA873,&H42A7

DATA &H2F07,&H487A,&HFE3A,&H487A

DATA &HFE3E,&H1F3C,&HFFFF,&H2F3C,0,0

DATA &H2F3C,0,0,&H2F3C,0,0,&HA954,&H201F

DATA &H2F04,&H2F03,&H487A,&HFE0C,&H3F3C

DATA 1,&HA9CE,&H42A7,&H487A,&HFF74

DATA &HA991,&H303A,&HFF6E,&H0C40,1

DATA &H66EE,&H2F07,&HA982,&H2F0B,&HA91F

DATA &H2F0B,&HA873,&H4E5E,&H4E75

FOR i=0 TO 19:READ CO(i):NEXT

FOR i=20 TO 189:CO(i)=0:NEXT

FOR i=190 TO 274:READ CO(i):NEXT:DE=191

'DE=début du code exécutable

M1\$="Ceci est le premier message"

M2\$="Ceci est le deuxième message"

M3\$="Ceci est le troisième message"

AI=VARPTR(CO(DE)):AI VARPTR(M1\$)

FOR i=0 TO 10

PRINT "Texte de remplissage / ";

PRINT "contrôle retour Basic

NEXT:PRINT

PRINT "Entrez un texte ci-dessous S.V.P.

INPUT z\$:PRINT:PRINT z\$

AI=VARPTR(CO(DE)):AI VARPTR(M2\$)

FOR i=0 TO 10:PRINT "Texte de remplissage / ";

PRINT "contrôle retour Basic

NEXT:PRINT

PRINT "Entrez un texte ci-dessous S.V.P.

INPUT z\$:PRINT:PRINT z\$

AI=VARPTR(CO(DE)):AI VARPTR(M3\$)

FOR i=0 TO 10

PRINT "Texte de remplissage / ";

PRINT "contrôle retour Basic

NEXT:PRINT

PRINT "Entrez un texte ci-dessous S.V.P.

INPUT z\$:PRINT:PRINT z\$

CO(0)=200:CO(1)=200:CO(2)=290:CO(3)=500

M1\$="Encore un message en boîte

AI=VARPTR(CO(DE)):AI VARPTR(M1\$):PRINT

PRINT "Encore une grande phrase à suivre

INPUT z\$

DATA 30,20,110,500,15,20,35,490

FOR i=0 TO 7:READ CO(i):NEXT

M2\$="Ce dernier message marque la fin de notre test
de présentation"

AI=VARPTR(CO(DE)):AI VARPTR(M2\$)

CO(0)=250:CO(2)=320:CO(8)=45:CO(10)=65

CLS:PRINT "Mais vous pouvez ";

PRINT "en rentrer un vous-même..."

INPUT M3\$:AI=VARPTR(CO(DE))

AI VARPTR(M3\$)

PRINT:PRINT M3\$

Fin:

GOTO Fin



Basicium... Le Basic enrichi

Gérard Michel

Apple II+, IIc

Cet utilitaire vous permet de :

Imprimer l'écran 40 ou 80
colonnes par "JH"

Nettoyer mémoire très
rapidement par "JF".

Gérer des messages, la réponse
étant contrôlée.

Gérer des masques 40 ou 80 colonnes.
Ces écrans, définis lors de la program-
mation, s'affichent par "JA".

Saisir toutes les variables d'un écran en
une seule commande "JI" en contrôlant
le type et la longueur.

Mais aussi, ne saisir que l'une des
variables affectées à un écran, les
afficher, les effacer, etc...

Tous les fichiers source sont sur la disquette (exemple de démonstration dans Pom's 13)

Disquette et documentation : 150,00 TTC franco (bon de commande page 74)

Gestion de comptes bancaires

J-L Bazanegue
C. Piard



Afin d'éviter les longues séances de pointages, raturages, surcharges à la réception des extraits de compte bancaire pour essayer de trouver un rapport entre le compte tenu chez soi et celui tenu par le banquier, voici un programme (presque) tout Basic qui peut rendre quelque service.

Un mac 128 ou 512 Ko et la version décimale du Basic Microsoft version 2.00 sont nécessaires, le lecteur externe étant facultatif.

Les principes

Ce programme peut gérer plusieurs groupes compte banque/carte de crédit. Dans les notes qui suivent, nous ne considérerons qu'un groupe.

Il est possible d'enregistrer les opérations suivantes :

- Dépenses par chèque ou prélèvement,
- Recettes sur le compte banque,
- Dépenses par carte de crédit,
- Virements carte de crédit,
- Pointage.

Chaque mois, le compte bancaire est débité des dépenses par carte bleue présentées par les bénéficiaires avant le 25 du mois précédent. Cette opération diminue la dette carte de crédit et diminue l'avoir du compte : c'est cette opération que nous nommons ici 'virement carte de crédit'.

Le pointage consiste à annoter les opérations enregistrées à la banque et chez soi. Ce pointage permet au programme lors de

l'édition des dernières opérations du compte de calculer le rapprochement, c'est à dire les soldes qui, théoriquement, apparaissent sur l'extrait de compte, tant sous la rubrique *compte principal* que sous *carte de crédit*.

Accessoirement, le programme calcule une ventilation des dépenses par type : Maison, loisirs, vêtements, vacances, transport... Ces types de dépenses sont éditables : à chaque groupe de comptes est attachée une ventilation que vous définirez au moment de la création. Vous ne pourrez plus revenir dessus après validation.

Ce programme, conçu pour un besoin particulier, ne gère pas les prélèvements automatiques, ne permet pas l'établissement de budget ou autre ; sa structure ne devrait toutefois pas poser de problème pour une personnalisation éventuelle.

Mode d'emploi

Première opération, initialisation d'un groupe compte/carte : c'est l'option **NOUVEAU** du menu **FICHER**.

La date demandée devra être postérieure à 1984, le nom du fichier est quelconque, peut être sous la forme BNCl 007665. Le solde banque est positif ou négatif, c'est selon ! La dette carte bleue est par convention forcément positive. Ces sommes devront rester dans les limites - raisonnables - de la double précision. Les noms des dix rubriques de ventilation des dépenses seront choisies de façon définitive à ce moment. Il est

préférable de mettre en numéro 1 la rubrique la plus fréquente : elle vous est proposée par défaut lors des saisies. Toutes les zones étant complétées, la validation par 'OK' vous est proposée.

Le fichier est alors créé en quelques instants sur le disque avec le type 'COMP' et enregistrement 1 la 'reprise de solde'.

Par la suite, pour travailler sur le fichier, il faudra choisir **OUVRI**R dans le menu **FICHER** : seuls les types 'COMP' s'afficheront.

Saisie est le premier item du menu **OPTION**. Pour ne pas s'y perdre, la dernière opération est rappelée en haut de l'écran (avec la notation des comptables : Débit correspond à une recette, Crédit à une dépense). Les soldes n'apparaissent pas en mode saisie, le bouton pointage reste invalide. Pour enregistrer effectivement votre opération, vous aurez au minimum indiqué la date, le premier libellé et le montant. La date sera sous la forme de six ou huit caractères avec un séparateur quelconque : 130985, 13/09/85, 13-09-85 sont des formes acceptées.

Consultation, deuxième item d'**OPTION** rend actifs les boutons au bas de l'écran.

Pointage, accessible depuis consultation, vous propose exclusivement les opérations non pointées ; celles-ci sont repérées dans un fichier spécial. Avant validation, vous pouvez modifier les libellés et le numéro d'opération.

Imprimer est accessible dans

le menu *FICHER*. Par défaut l'impression se fait entre la plus ancienne opération non enregistrée à la banque et la dernière ligne saisie. Au bas de l'état sont indiqués les soldes théoriques à la banque puis la ventilation des lignes de dépenses imprimées. Il

est à remarquer que le libellé 2 ne figure pas sur l'état.

Les limitations de ce programme sont les suivantes :

- 1000 opérations par an c'est à dire environ 100 Ko (modifiable ligne Ncomptes),

- 100 opérations non pointées (modifiable ligne Npoint),
- 900 millions de francs, pour rêver...



Programme Comptes

```
DEFINT A-Z
DIM C1(7),TV(14),EF$(14),R(14,3),TAB
L(100),VENT$(10):US$="#####.##"
DATA &h4E56,0,&h42A7,&h3F2E,&h8,
&hA9B9,&h2057,&h2E90,&hA851,&h4
E5E,&h4E75
DATA 1,Fichiers,1,Nouveau,1,Ouvrir,0,
Fermer,0,Imprimer,1,Quitter
'Sup fenêtres
DATA 0,Options,2,Saisie,1,Consultatio
n,1,pointage
DATA &h42A7,&hA924,&h2E1F,&h6706
,&h2F07,&hA916,&h60F2,&h4E75
'Sup Menus
DATA &h4E56,&h0000,&h3F2E,&h0008
,&hA936,&hA937,&h4E5E,&h4E75
'coordonnées
Coo:
DATA 8,8,112,288,120,8,272,288
DATA Date,Nom de fichier,Montant B
anque,Dû Carte bleue
Coo2:
DATA 72,344,280,488,168,8,208,328
DATA 216,144,232,328,240,144,256,32
8
CooS:
DATA 72,73,146,88,224,73,328,88,72,
97,328,112,72,121,328,136,72,145,17
6,160
FOR i=0 TO 10:READ C2(i):NEXT
FOR i=0 TO 5:READ N,C$:MENU 4
,I,N,C$:NEXT
FOR i=0 TO 3:READ N,C$:MENU 5,
I,N,C$:NEXT
GOSUB S1:A!
GOSUB S1:FOR i=0 TO 4:A! I:NEX
T
ON MENU GOSUB Me0:MENU ON
B1:
GOTO B1
Me0:
IF MENU(0)=5 THEN Me1 ELSE O
N MENU(1) GOSUB Me01,Me02,M
e03,Me04,Me05:RETURN
Me01:
WINDOW 1,,(104,24)-(400,336),-3:TE
XTFONT 0:TEXTSIZE 12:TEXTM
ODE 1:FOR i=0 TO 3:C1(i)=&HAA5
5:NEXT:PENPAT VARPTR(C1(0))
RESTORE Coo:GOSUB S1:FOR i=0
TO 4 STEP 4:FRAMERECT VA
RPTR(C1(i)):NEXT
N=0:FOR i=32 TO 104 STEP 24:N=
N+1:EDIT FIELD N,"",(176,i-15)-(28
```

```
0,i):R(N,0)=176:R(N,1)=i-15:R(N,2)=28
0:R(N,3)=i
READ C$:MOVETO 16,i:PRINT C$:
GETPEN VARPTR(C1(0)):LINE(C1(
1),i)-(168,i):NEXT
FOR i=32 TO 176 STEP 144:FOR
J=153 TO 265 STEP 24:N=N+1:ED
IT FIELD N,"",(i,J)-(i+104,J+15):R(N
,0)=i:R(N,1)=J:R(N,2)=i+104:R(N,3)=J+
15
MOVETO i-22,J+12:PRINT USING"#
":N-4:NEXT:NEXT
MOVETO 16,136:PRINT "Noms de r
ubrique pour ventilation":MOVETO 1
6,144:LINETO 280,144
BUTTON 1,0,"OK",(64,280)-(128,304),1
:BUTTON 2,1,"Annuler",(168,280)-(23
2,304),1:EDIT FIELD 1:EF=1
FOR i=1 TO 14:TV(i)=0:NEXT:ON D
IALOG GOSUB Di:IALOG ON
RETURN
Me02:
F$=FILES$(1,"COMP"):IF f$="" THEN
MENU:RETURN
OPEN"R",1,F$,112:F$=LEFT$(F$,LEN(
F$)-2)+".P":OPEN"R",2,F$,2:F$=LEFT
$(F$,LEN(F$)-2)+".V":OPEN"R",3,F$,
25
Me022:
FIELD 1,6 AS zDA$,11 AS zNum$,32
AS zLib1$,32 AS zLib2$,8 AS zMo
n$,8 AS zSol$,8 AS zCB$,2 AS zC
ode$,2 AS zPoin$,2 AS zVent$
FIELD 2,2 AS zNume$
FIELD 3,25 AS zN$
GET 2,1:Num=CVI(zNume$)
WINDOW 1,,(8,24)-(504,336),3:TEXT
FONT 0:TEXTSIZE 12:TEXTMOD
E 1
LINE(8,8)-(488,64),B:FOR i=0 TO 3:
C1(i)=&HAA55:NEXT:PENPAT VAR
PTR(C1(0)):MOVETO 16,32:LINETO
480,32:RESTORE Coo2:FOR i=0
TO 7:READ C1(i):NEXT:FRAMER
ECT VARPTR(C1(0)):FRAMERECT
VARPTR(C1(4))
FOR i=0 TO 7:READ c1(i):NEXT:F
RAMERECT VARPTR(C1(0)):FRA
MERECT VARPTR(C1(4))
MOVETO 353,87:PRINT"Ventilation":
MOVETO 352,96:LINETO 480,96:F
OR i=0 TO 9:BUTTON i+1,1,"",(353
,109+16*i)-(367,121+16*i),3:GET 3,i+
1:MOVETO 376,120+16*i:PRINT z
N$:EF$(i+1)=zN$:NEXT:BUTTON 1,
2
GOSUB Aff:PENMODE 11:C1(0)=144
:C1(1)=240:C1(2)=162:C1(3)=304:C1(4)
=216:C1(5)=8:C1(6)=258:C1(7)=120:P
```

```
AINRECT VARPTR(c1(0)):PAINT
RECT VARPTR(c1(4))
BUTTON 11,0,"",(309,145)-(323,160),3:
n=11:FOR i=133 TO 309 STEP 17
6:FOR j=172 TO 188 STEP 16:n=n
+1:BUTTON n,1,"",(i,j)-(i+14,j+14),3
:NEXT:NEXT:BUTTON 13,2
GET 1,Num+1:GOSUB RAPPEL
GOSUB EField:BUTTON 16,0,"Premie
r",(8,264)-(112,280):BUTTON 17,0,"1
0 en arrière",(120,264)-(224,280):BU
TTON 18,0,"Précédent",(232,264)-(33
6,280)
BUTTON 19,0,"Dernier",(8,288)-(112,30
4):BUTTON 20,0,"10 en avant",(120
,288)-(224,304):BUTTON 21,0,"Suiva
nt",(232,288)-(336,304):BUTTON 22,
0,"Valider",(344,288)-(488,304)
DM=1:MENU:MENU 4,0,1:MENU 4,1,
0:MENU 4,2,0:FOR i=3 TO 5:ME
NU 4,i,1:NEXT:MENU 5,0,1:MENU 5
,3,0
EFS=1:EF=11:RESTORE CooS:FOR i
=1 TO 5:READ R(i,0),R(i,1),R(i,2),R
(i,3):NEXT:FOR i=1 TO 3:TV(i)=0:N
EXT:LSET zVent$=MKI$(1):LSET
zPoin$=MKI$(0):LSET zCode$=MKI
$(2):LSET zLib2$="" :LSET zNum$=""
:ON DIALOG GOSUB DSaisie:DI
ALOG ON:RETURN
Me03:
GOSUB Me03b
CLOSE:RETURN
Me03b:
WINDOW CLOSE 1:GOSUB MMenu
:GOSUB FQ:RETURN
MMenu:
MENU 4,1,1:MENU 4,2,1:MENU 4,3,
0:MENU 4,4,0:MENU 4,5,1:MENU
5,0,0:MENU 5,1,2:MENU 5,2,1:ME
NU 5,3,0:RETURN
FQ:
IF DM=3 THEN GOSUB MJPoint
RETURN
Me04:
DIALOG OFF:GET 2,1:NDL=CVI(zN
ume$)+1:GET 2,2:NNP=CVI(zNume$)
:GET 2,NNP+2:NDL1=CVI(zNume$):
GET 2,3:NPL=CVI(zNume$):NPL1=N
PL
IF NNP=0 THEN NPL=NDL-10
IF NPL<1 THEN NPL=1
GOSUB Me03b:WINDOW 1,,(172,127
)-(348,215),-3:TEXTFONT 0:MOVE
TO 16,24:PRINT"Première ligne":M
OVETO 16,48:PRINT"Dernière ligne"
EDIT FIELD 1,STR$(NPL),(120,12)-(
160,27),1,3
```




```

EDIT FIELD 2,STR$(NDL-1),(120,36)
-(160,51),1,3
EF=1:BUTTON 1,1,"OK",(16,64)-(82,79)
):BUTTON 2,1,"Annuler",(94,64)-(160,79):ON DIALOG GOSUB DImp:DIALOG ON

```

```

DImp:
N=DIALOG(0):IF N=1 THEN BImp ELSE IF N=2 THEN SImp ELSE IF N=6 OR N=7 THEN RTImp ELSE RETURN

```

```

BImp:
IF DIALOG(1)=2 THEN WINDOW CLOSE 1:GOSUB Me03:CLOSE:RETURN

```

```

xNPL=VAL(EDIT$(1)):xNDL=VAL(EDIT$(2))
IF xNPL>xNDL OR xNPL<0 OR xNDL>NDL-1 THEN BEEP:EDIT FIELD 1:RETURN

```

```

NPL=xNPL+1:NDL=xNDL+1:IF NPL>NPL+1 OR NDL<NDL+1 THEN drap=0 ELSE drap=-1

```

```

GOTO Impres

```

```

SImp:
IF DIALOG(2)=EF THEN RETURN ELSE EF=DIALOG(2):GOTO DImp

```

```

RTImp:
IF EF=2 THEN EF=1 ELSE ef=2

```

```

DImpS:
EDIT FIELD EF:RETURN

```

```

Me05:
GOSUB FQ:CLOSE:SYSTEM

```

```

Me1:
ON MENU(1) GOSUB Me11,Me12,Me13:RETURN

```

```

Me11:
LINE(9,9)-(487,30),30,BF:LINE(9,34)-(487,63),30,BF

```

```

GET 2,1:Num=CVI(zNume$):GET 1,Num+1:GOSUB rappel

```

```

PENMODE 8:C1(0)=144:C1(1)=240:C1(2)=162:C1(3)=304:C1(4)=216:C1(5)=8:C1(6)=258:C1(7)=120:PENMODE 11:PAINTRECT VARPTR(C1(0)):PAINTRECT VARPTR(C1(4))

```

```

LINE(145,217)-(326,230),30,BF:LINE(145,241)-(326,254),30,BF

```

```

RESTORE coo2:FOR i=0 TO 7:READ c1(i):NEXT:FOR i=0 TO 7:READ c1(i):NEXT:PENMODE 8:FRAME RECT VARPTR(C1(0)):FRAME RECT VARPTR(C1(4)):PENMODE 11:FOR i=15 TO 22:BUTTON 1,0:NEXT:FOR i=1 TO 10:BUTTON 1,1:NEXT:BUTTON 1,2:BUTTON 11,0:FOR i=12 TO 15:BUTTON 1,1:NEXT:BUTTON 13,2

```

```

DM=1:MENU:MENU 5,1,2:MENU 5,2,1:MENU 5,3,0:GOSUB Efield

```

```

EFS=1:EF=11:FOR i=1 TO 3:TV(I)=0:NEXT:LSET zVent$=MKI$(1):LSET zPoin$=MKI$(0):LSET zCode$=MKI$(2):LSET zLib2$="":LSET zNum$="":ON DIALOG GOSUB DSaisie:DIALOG ON:RETURN

```

```

Me12:
DIALOG OFF:IF DM=2 THEN RETURN ELSE IF DM=3 GOTO Me123

```

Me12S:

```

DM=2:C1(0)=9:C1(1)=9:C1(2)=30:C1(3)=487:C1(4)=34:C1(5)=9:C1(6)=63:C1(7)=487:PENMODE 11:PAINTRECT VARPTR(C1(0)):PAINTRECT VARPTR(C1(4))

```

```

FOR i=1 TO 15:BUTTON 1,1:NEXT:FOR i=1 TO 5:EDIT FIELD CLOSE I:NEXT

```

```

LINE(223,72)-(329,89),B:LINE(71,96)-(329,113),B:LINE(71,120)-(329,137),B:LINE(71,144)-(177,161),B:LINE(71,72)-(147,89),B

```

```

RESTORE coo2:FOR i=0 TO 7:READ C1(i):NEXT:FOR i=0 TO 7:READ C1(i):NEXT:PENMODE 14:FRAME RECT VARPTR(C1(0)):FRAME RECT VARPTR(C1(4))

```

```

MOVETO 16,228:PRINT"Solde Banque":MOVETO 16,252:PRINT"Dû Carte bleue":MOVETO 242,159:PRINT"Pointage"

```

```

BUTTON 22,0:FOR i=16 TO 18:BUTTON 1,1:NEXT:MENU 5,1,1:MENU 5,2,2:MENU 5,3,1:GET 2,1:Num=CVI(zNume$)+1:NumF=Num:GET 2,2:Nomb=CVI(zNume$):GOSUB AffCons:ON DIALOG GOSUB consult:DIALOG ON:RETURN

```

```

Me123:
GOSUB MJPoint:GOSUB vide:GOTO Me12S

```

```

MJPoint:
DM=2:FOR i=1 TO Nombre:LSET zNume$=MKI$(TABL(I)-1):PUT 2,1+2:NEXT

```

```

LSET zNume$=MKI$(Nombre):PUT 2,2:RETURN

```

```

Me13:
IF Nombre=0 THEN MENU 5,3,0:RETURN

```

```

DM=3:DIALOG OFF

```

```

GOSUB Vide

```

```

FOR i=16 TO 21:BUTTON 1,1:NEXT:FOR i=16 TO 17:BUTTON 1,0:BUTTON 1+3,0:NEXT

```

```

MENU:FOR i=0 TO 2:MENU 5,1+1,i:NEXT

```

```

FOR i=1 TO Nombre:GET 2,1+2:TAB L(I)=CVI(zNume$)+1:NEXT:II=1

```

```

GOSUB AffPoint

```

```

BUTTON 18,0:IF nombre=1 THEN BUTTON 21,0

```

```

ON DIALOG GOSUB Pointage:DIALOG ON:RETURN

```

Pointage:

```

N=DIALOG(0):IF N=1 THEN PoinBouton ELSE IF N=2 THEN PoinSour ELSE PointRT

```

RETURN

PoinBouton:

```

N=DIALOG(1):IF N=11 THEN BPoint ELSE IF N=18 THEN prec ELSE IF N<21 THEN RETURN

```

```

IF N=21 THEN Suiv ELSE Valide

```

BPoint:

```

IF BUTTON(11)=1 THEN BUTTON 11,2:BUTTON 22,1 ELSE BUTTON 11,1:BUTTON 22,0
RETURN

```

prec:

```

BUTTON 21,1:II=II-1:IF II=1 THEN BUTTON 18,0
GOSUB vide:GOSUB AffPoint:RETURN

```

Suiv:

```

BUTTON 18,1:II=II+1:IF II=Nombre THEN BUTTON 21,0
GOSUB vide:GOSUB AffPoint:RETURN

```

Valide:

```

LSET zLib1$=EDIT$(2):LSET zLib2$=EDIT$(3):LSET zNum$=EDIT$(1):LSET zPoin$=MKI$(-1):PUT 1,TABL(II)

```

Nombre=Nombre-1

```

IF nombre=0 THEN GOSUB vide:FOR i=1 TO 22:BUTTON 1,0:NEXT:EDIT FIELD 3,"",(72,121)-(328,136):EDIT FIELD 2,"",(72,97)-(328,112):EDIT FIELD 1,"",(224,73)-(328,88):RETURN

```

```

FOR i=II TO nombre:SWAP TABL(I),TABL(II+1):NEXT

```

```

IF II=Nombre+1 THEN BUTTON 21,0:II=II-1

```

```

IF II=1 THEN BUTTON 18,0
GOSUB vide:GOSUB AffPoint:RETURN

```

PoinSour:

```

IF DIALOG(2)=EF THEN RETURN ELSE NEF=DIALOG(2)

```

GOTO DPoint

PointRT:

```

IF EF=3 THEN NEF=1 ELSE NEF=EF+1

```

DPoint:

```

EF$(EF+10)=EDIT$(EF)

```

```

IF EF>1 THEN Point2

```

Point1b:

```

IF WIDTH(EF$(11))>96 THEN EF$(11)=LEFT$(EF$(11),LEN(EF$(11))-1):GOTO Point1b ELSE EDIT FIELD 1,EF$(11),(224,73)-(328,88):EF=NEF:EDIT FIELD EF:RETURN

```

Point2:

```

IF WIDTH(EF$(EF+10))>240 THEN EF$(EF+10)=LEFT$(EF$(EF+10),LEN(EF$(EF+10))-1):GOTO Point2 ELSE EDIT FIELD EF,EF$(EF+10),(72,97)+(24*(EF-2))-(328,112)+(24*(EF-2)):EF=NEF:EDIT FIELD EF:RETURN

```

Vide:

```

LINE(72,145)-(176,160),30,BF:LINE(72,73)-(146,88),30,BF

```

```

LINE(145,217)-(326,230),30,BF:LINE(145,241)-(326,254),30,BF

```

```

FOR i=1 TO 15:BUTTON 1,1:NEXT

```

RETURN

AffPoint:

```

GET 1,TABL(II):EF=1

```

```

EDIT FIELD 3,zLib2$,(72,121)-(328,136):ef$(13)=zLib2$

```

```

EDIT FIELD 2,zLib1$,(72,97)-(328,112):ef$(12)=zLib1$

```

```

EDIT FIELD 1,zNum$,(224,73)-(328,88):ef$(11)=zNum$

```

```

MOVETO 80,85:PRINT LEFT$(zDA$,2)+"/"+MID$(zDA$,3,2)+"/"+RIGHT$(zDA$,2)

```



```

MOVETO 80,157:PRINT USING U$;
CVD(zMon$)
MOVETO 220,229:PRINT USING U
$;CVD(zSol$):MOVETO 220,253:PR
INT USING U$;CVD(zCB$)
code=CVI(zCode$):BUTTON code+11,
2:IF CVI(zPoin$) THEN BUTTON
11,2
IF CVI(zVent$) THEN BUTTON CVI
(zVent$),2
BUTTON 22,0
RETURN

```

```

Consult:
N=DIALOG(0):N=DIALOG(1):IF N<16
THEN RETURN
ON (N-15) GOTO C1,C10M,CP,CD,C1
OP,CS
C1:
NumF=1:GOTO SConsult
C10M:
IF (NumF-10)<1 THEN NumF=1 ELS
E NumF=NumF-10
GOTO SConsult
CP:
NumF=NumF-1:GOTO SConsult
CD:
NumF=Num:GOTO SConsult
C10P:
IF (NumF+10)>Num THEN NumF=Num
ELSE NumF=NumF+10
GOTO SConsult
CS:
NumF=NumF+1:GOTO SConsult

```

```

SConsult:
FOR I=1 TO 15:BUTTON I,1:NEXT
LINE(225,75)-(327,87),30,BF:LINE(73,
98)-(327,111),30,BF:LINE(73,122)-(32
7,135),30,BF:LINE(73,146)-(175,159),
30,BF:LINE(73,74)-(145,87),30,BF
LINE(145,217)-(326,230),30,BF:LINE(1
45,241)-(326,254),30,BF
IF num=1 THEN FOR I=16 TO 21:B
UTTON I,0:NEXT:GOTO SSConsul
t
FOR I=16 TO 21:BUTTON I,1:NEXT
IF numF=1 THEN FOR i=16 TO 18:
BUTTON i,0:NEXT
IF numF = num THEN FOR i = 19
TO 21:BUTTON I,0:NEXT
IF (numF-10)<1 THEN BUTTON 17,0
IF (numF+10)>num THEN BUTTON 20
,0

```

```

SSConsult:
GOSUB AffCons:RETURN
EField:
EDIT FIELD 2,"",(224,73)-(328,88):E
DIT FIELD 3,"",(72,97)-(328,112):E
DIT FIELD 4,"",(72,121)-(328,136):E
DIT FIELD 5,"",(72,145)-(176,160)
EDIT FIELD 1,LEFT$(zDA$,2)+"/"+M
ID$(zDA$,3,2)+"/"+RIGHT$(zDA$,2),(
72,73)-(146,88):RETURN

```

```

VentOff:
C1(0)=112:C1(1)=376:C1(2)=274:C1(3)=4
87:PAINTRECT VARPTR(c1(0)):F

```

```

OR I=1 TO 10:BUTTON I,0:NEXT:
LSET zVent$=MKI$(0):RETURN

```

```

Aff:
MOVETO 9,87:PRINT"Date":MOVET
O 168,87:PRINT"Numéro":MOVETO
9,111:PRINT"Lib. 1":MOVETO 9,1
35:PRINT"Lib. 2":MOVETO 9,159:P
RINT"Montant":MOVETO 16,228:PR
INT"Solde Banque":MOVETO 16,25
2:PRINT"Dû Carte bleue
MOVETO 16,183:PRINT"Recette Ban
que":MOVETO 16,199:PRINT"Dépe
nse Banque":MOVETO 165,183:PRI
NT"Dépense Carte Bleue":MOVETO
165,199:PRINT"Virement Carte Ble
ue":MOVETO 242,159:PRINT"Point
age
RETURN

```

```

Rappel:
N=0:FOR I=1 TO LEN(F$):IF MID$(
F$,I,1)="-" THEN N=N+1
NEXT:IF N THEN F$=RIGHT$(F$,LE
N(F$)-N)
MOVETO 16,24:PRINT "Fichier "" L
EFT$(F$,LEN(F$)-2) "" - Rappel de
l'enregistrement précédent : " Num
MOVETO 16,44:PRINT LEFT$(zDA$,
2)+"/"+MID$(zDA$,3,2)+"/"+RIGHT$(
zDA$,2)
MOVETO 93,44:PRINT "N" " zNum$:
MOVETO 214,44:PRINT zLib1$
N=CVI(zCode$):V1#=0:V2#=0:V3#=0:O
N N-1 GOTO SN2,SN3,SN4
V1#=CVD(zMon$):GOTO SN0
SN2:
V2#=CVD(zMon$):GOTO SN0
SN3:
V3#=CVD(zMon$):GOTO SN0
SN4:
V2#=CVD(zMon$):V3#=-CVD(zMon$)
SN0:
MOVETO 16,59:PRINT USING"Débit
Bq :#####.## Crédit Bq :##
#####.## CB :#####.##":V
1#,V2#,V3#
RETURN

```

```

Di:
N=DIALOG(0):IF N=1 THEN Bouton
ELSE IF N=2 THEN Dsourir ELSE
IF N=6 OR N=7 THEN TabRetur
n ELSE RETURN

```

```

Bouton:
IF DIALOG(1)=1 THEN BoutonOK
MENU:WINDOW CLOSE 1:RETUR
N
BoutonOK:
MENU:DIALOG OFF:AI=VARPTR(C
2(0)):AI 4:OPEN"R",1,EF$(2),25:NA
ME EF$(2) AS EF$(2)+".V"
FIELD 1,25 AS zN$
FOR I=1 TO 10:LSET zN$=EF$(I+4):
PUT 1,I:NEXT
CLOSE
OPEN"R",1,EF$(2),112:NAME EF$(2)
AS EF$(2)+".D",COMP"
FIELD 1,6 AS zDA$,11 AS zNum$,3

```

```

2 AS zLib1$,32 AS zLib2$,8 AS z
Mon$,8 AS zSol$,8 AS zCB$,2 AS
zCode$,2 AS zPoin$,2 AS zVent$
Ncomptes:
FOR I=1 TO 1001:PUT #1,I:NEXT
LSET zDA$=JJ$+MM$+AA$:LSET zN
um$="" :LSET zLib1$="Reprise de sol
des":LSET zLib2$=""
V#=-VAL(ef$(3)):IF v#<0 THEN V#=-A
BS(v#):n=2 ELSE n=1
LSET zMon$=MKD$(V#):LSET zSol$
=MKD$(VAL(EF$(3)))
LSET zCB$=MKD$(VAL(EF$(4))):LSE
T zCode$=MKI$(N):LSET zPoin$=M
KI$(-1):LSET zVent$=MKI$(0)
PUT 1,1
CLOSE
OPEN"R",1,EF$(2),2:NAME ef$(2) AS
ef$(2)+".P"
FIELD 1,2 AS zP$:LSET zP$=MKI$
(0):PUT 1,1
Npoint:
FOR i=2 TO 102:PUT 1,i:NEXT:CL
OSE
WINDOW CLOSE 1:INITCURSOR:
DIALOG ON:RETURN

```

```

Dsourir:
IF DIALOG(2)=EF THEN RETURN
ELSE NEF=DIALOG(2):GOTO Trai
t

```

```

TabReturn:
IF EF=14 THEN NEF=1 ELSE NEF=
EF+1

```

```

Trait:
IF EF>4 THEN Ventil ELSE ON EF-
1 GOTO Nom,MB,CB
EF$(1)=EDIT$(1):IF LEN(EF$(1))=0 T
HEN TV(1)=0:GOTO Dsuivant ELS
E GOSUB Vdate
IF DE THEN TV(1)=0:GOTO Erreur
ELSE TV(1)=-1:GOTO Dsuivant

```

```

Nom:
EF$(2)=EDIT$(2):IF LEN(EF$(2))=0 T
HEN TV(2)=0:GOTO Dsuivant ELS
E GOSUB VLongP
TV(2)--1:GOTO Dsuivant

```

```

MB:
EF$(3)=EDIT$(3):IF LEN(EF$(3))=0 T
HEN TV(3)=0:GOTO Dsuivant ELS
E GOSUB VValN
IF DE THEN TV(3)=0:GOTO Erreur
ELSE TV(3)=-1:GOTO Dsuivant

```

```

CB:
EF$(4)=EDIT$(4):IF LEN(EF$(4))=0 T
HEN TV(4)=0:GOTO Dsuivant ELS
E GOSUB VValP
IF DE THEN TV(4)=0:GOTO Erreur
ELSE TV(4)=-1:GOTO Dsuivant

```

```

Ventil:
EF$(EF)=EDIT$(EF):IF LEN(EF$(EF))=
0 THEN TV(EF)=0:GOTO Dsuivant
ELSE GOSUB VLongP
TV(EF)--1:GOTO Dsuivant

```

```

Dsuivant:
EDIT FIELD EF,EF$(EF),(R(EF,0),R(E
F,1))-(R(EF,2),R(EF,3)):EF=NEF:EDIT
FIELD EF
DE=-1:FOR I=1 TO 14:IF TV(I)=0 TH

```




```
EN DE=0
NEXT:IF DE THEN IF BUTTON(1)=
0 THEN BUTTON 1,1:RETURN
IF DE=0 THEN IF BUTTON(1)=1 T
HEN BUTTON 1,0
RETURN
```

```
Erreur:
BEEP:BEEP:EDIT FIELD EF,EF$(E
F),(R(EF,0),R(EF,1))-(R(EF,2),R(EF,3)):
RETURN
```

```
Me05:
MENU RESET:END
```

```
Vdate:
DE=0:IF LEN(EF$(EF))=6 THEN N=-1
ELSE IF LEN(EF$(EF))=8 THEN
N=0 ELSE VdateEr
JJ$=LEFT$(EF$(EF),2):IF VAL(JJ$)>3
1 THEN VdateEr
IF N THEN MM$=MID$(EF$(EF),3,2)
ELSE MM$=MID$(EF$(EF),4,2)
IF VAL(MM$)>12 THEN VdateEr
AA$=RIGHT$(EF$(EF),2):IF VAL(AA$)
<85 THEN VdateEr
EF$(EF)=JJ$+"/"+MM$+"/"+AA$:RETUR
N
VdateEr:
DE=-1:RETURN
```

```
VLongP:
L=WIDTH(EF$(EF)):IF L<96 THEN R
ETURN ELSE EF$(EF)=RIGHT$(E
F$(EF),LEN(EF$(EF))-1):GOTO VLon
gP
```

```
VValN:
DN=-1:GOTO VVal
VValP:
DN=0
VVal:
N=0:L=LEN(EF$(EF)):FOR I=1 TO L:C
$=MID$(EF$(EF),I,1):IF C$="?" OR
C$="," OR C$=";" OR C$="." THEN
MID$(EF$(EF),I,1)="":N=N+1
NEXT:IF N THEN IF L>(N+2) THEN
EF$(EF)=LEFT$(EF$(EF),N+2)
N$=VAL(EF$(EF)):N$=INT(N$*100+.5)/1
00:EF$(EF)=STR$(N$):IF LEFT$(EF
$(EF),1)=" " THEN EF$(EF)=RIGHT
$(EF$(EF),LEN(EF$(EF))-1)
IF N=0 OR N=L THEN EF$(EF)=EF$(
EF)+".00"
L=LEN(EF$(EF)):IF (N+2)-L=1 THEN
EF$(EF)=EF$(EF)+".0"
IF DN THEN IF N#>999999999.99#
OR N#<-999999999.99# THEN DE=-1
:RETURN ELSE DE=0:RETURN
IF N#>999999999.99# OR N#<0 THE
N DE=-1:RETURN ELSE DE=0:RE
TURN
```

```
S1:
FOR I=0 TO 7:READ C1(I):NEXT:A!
=VARPTR(C1(0)):RETURN
```

```
DSaisie:
N=DIALOG(0):IF N=1 THEN SBouton
ELSE IF N=2 THEN SDsouris EL
```

```
SE IF N=6 OR N=7 THEN STabRe
turn ELSE RETURN
SBouton:
N=DIALOG(1):IF N<11 THEN VentilS
ELSE IF N=22 THEN ValidS
FOR I=12 TO 15:BUTTON I,1:NEXT
:BUTTON N,2:NN=BUTTON(1)
IF NN=0 THEN IF N=13 OR N=14 T
HEN FOR I=0 TO 9:BUTTON I+1,
1:MOVETO 376,120+I*16:PRINT E
F$(I+1):NEXT:BUTTON 1,2:GOTO
SBoutonS
IF N=12 OR N=15 THEN GOSUB V
entOff
SBoutonS:
Code=N-11:GOTO SDsouris2
```

```
VentilS:
FOR I=1 TO 10:BUTTON I,1:NEXT:
BUTTON N,2:Ventil=N:GOTO SDSo
uris2
```

```
ValidS:
IF BUTTON(13)=2 THEN code=2
IF BUTTON(1)=2 THEN LSET zVen
t$=MKI$(1)
IF BUTTON(1)=1 THEN LSET zVen
t$=MKI$(Ventil) ELSE IF BUTTON
(1)=0 THEN LSET zVent$=MKI$(0)
Cbleu$=CVD(zCB$):Solde$=CVD(zSol
$):LSET zCode$=MKI$(code):LSET
zMon$=MKD$(VAL(montant$)):ON c
ode-1 GOTO code2,code3,code4
LSET zSol$=MKD$(VAL(Montant$)+S
olde$):GOTO EnrS
code2:
LSET zSol$=MKD$(Solde$-VAL(Mont
ant$)):GOTO EnrS
code3:
LSET zCB$=MKD$(Cbleu$+VAL(Mon
tant$)):GOTO EnrS
code4:
LSET zCB$=MKD$(Cbleu$-VAL(Mon
tant$)):LSET zSol$=MKD$(Solde$-V
AL(Montant$))
```

```
EnrS:
GET 2,2:NEnrNP=CVI(zNume$)
LSET zNume$=MKI$(Num+1):PUT 2,
1
LSET zNume$=MKI$(NEnrNP+1):PUT
2,2
LSET zNume$=MKI$(Num+1):PUT 2,
NEnrNP+3
PUT 1,Num+2
GET 2,1:Num=CVI(zNume$):FOR I=0
TO 9:BUTTON I+1,1
MOVETO 376,120+I*16:PRINT ef$(I+
1):NEXT:BUTTON 1,2
BUTTON 22,0:LINE(9,9)-(487,30),30,
BF:LINE(9,34)-(487,63),30,BF
BUTTON 12,1:BUTTON 13,2:BUUTO
N 14,1:BUTTON 15,1
GET 1,Num+1:GOSUB rappel
GOSUB Efield:EFS=1:EF=11:FOR I=1
TO 3:TV(I)=0:NEXT:LSET zLib2$=
"":LSET zNum$="":LSET zVent$=M
KI$(1):LSET zPoin$=MKI$(0):LSET
zCode$=MKI$(2):RETURN
```

```
SDsouris2:
```

```
NEFS=DIALOG(2):DV=0:GOTO TraitS
SDsouris:
IF DIALOG(2)=EFS THEN RETURN
ELSE NEFS=DIALOG(2):DV=-1:G
OTO TraitS
STabReturn:
IF EFS=5 THEN NEFS=1 ELSE NEF
S=EFS+1
TraitS:
ON EFS-1 GOTO NumS,Lib1S,Lib2S,
MontantS
EF$(EF)=EDIT$(1):IF LEN(ef$(EF))=0
THEN TV(1)=0:GOTO DsuivantS:EL
SE GOSUB Vdate
IF DE THEN TV(1)=0:GOTO ErreurS
ELSE TV(1)=1:LSET zDA$=LEFT
$(EF$(EF),2)+MID$(EF$(EF),4,2)+RI
GHT$(EF$(EF),2):GOTO DsuivantS
NumS:
EF$(EF)=EDIT$(2)
NumSb:
IF LEN(EF$(EF))>11 THEN EF$(EF)=
LEFT$(EF$(EF),LEN(EF$(EF))-1):GO
TO NumSb ELSE LSET zNum$=EF
$(EF):GOTO DsuivantS
Lib1S:
EF$(EF)=EDIT$(3):IF LEN(EF$(EF))=0
THEN TV(2)=0:GOTO DsuivantS
TV(2)=1
Lib1Sb:
IF WIDTH(EF$(EF))>240 THEN EF$(
EF)=LEFT$(EF$(EF),LEN(EF$(EF))-1)
:GOTO Lib1Sb ELSE LSET zLib1
$=EF$(EF):GOTO DsuivantS
Lib2S:
EF$(EF)=EDIT$(4)
Lib2Sb:
IF WIDTH(EF$(EF))>240 THEN EF$(E
F)=LEFT$(EF$(EF),LEN(EF$(EF))-1)
:GOTO Lib2Sb ELSE LSET zLib2$=E
F$(EF):GOTO DsuivantS
MontantS:
EF$(EF)=EDIT$(5):IF LEN(EF$(EF))=0
THEN TV(3)=0:GOTO DsuivantS
ELSE GOSUB VValP
IF DE THEN TV(3)=0:GOTO ErreurS
ELSE TV(3)=1:Montant$=EF$(EF):
GOTO DsuivantS
DsuivantS:
EDIT FIELD EFS,EF$(EF),(R(EFS,0),
R(EFS,1))-(R(EFS,2),R(EFS,3)):IF DV
THEN EFS=NEFS:EDIT FIELD EF
S
GOSUB VerifS:RETURN
ErreurS:
BEEP:BEEP:EDIT FIELD EFS,EF$(
EF),(R(EFS,0),R(EFS,1))-(R(EFS,2),R(
EFS,3)):RETURN
VerifS:
DE=-1:FOR I=1 TO 3:IF TV(I)=0 TH
EN DE=0
NEXT:IF DE THEN IF BUTTON(22)
=0 THEN BUTTON 22,1:RETURN
IF DE=0 THEN IF BUTTON(22)=1 T
HEN BUTTON 22,0
RETURN
```




Fichier BNCI 007665 - Rappel de l'enregistrement précédent : 6

19/08/85 N° 8248253

Editions MEU Recueil Pom's 2

Débit Bq : 0.00 Crédit Bq : 0.00 CB : 140.00

Date 22/08/85 Numéro 6443997

Lib. 1 Photo Shop

Lib. 2 Bague Macro

Montant 766.00

Pointage ☐Recette Banque ☐ Dépense Carte Bleue ☐Dépense Banque ☒ Virement Carte Bleue ☐

Solde Banque

Où Carte bleue

Premier

10 en arrière

Précédent

Dernier

10 en avant

Suivant

Ventilation

- ☐ Loyer
☐ Alimentation
☐ Vêtements
☐ Loisirs
☐ Vacances
☐ Impôts
☐ Ordinateur
☒ Photographie
☐ Moto
☐ Divers

Valider

Impres:

WIDTH=LPT1: ",136

Page=0:TD#=0:TC#=0:TCB#=0:SBT#=0:S
CT#=0

FOR i=1 TO 10:VENT#(i)=0:NEXT

GOSUB entete

FOR i=NPL TO NDL

GET 1,I

MT#=CVD(zMon\$):So#=CVD(zSol\$):CB

#=CVD(zCB\$):Co=CVI(zCode\$)

Li=Li+1

IF Li>55 THEN GOSUB totaux:LPR

INT CHR\$(12)::GOSUB entete

GOSUB Ligne

NEXT

GOSUB Totaux

IF NOT DRAP THEN LPRINT "Editi

on insuffisante pour calcul du rapp

ochement":GOTO ici

LPRINT"SOLDES THEORIQUES SUR

EXTRAIT BANCAIRE :

LPRINT TAB(30):USING U\$;SBT#+S

o#::LPRINT" en compte principal

LPRINT TAB(30):USING U\$;SCT#+C

B#::LPRINT" en carte bleue

ici:

LPRINT CHR\$(12)

LPRINT "VENTILATION DES DEPENS

ES :

FOR i=1 TO 10:LPRINT EF\$(i);TAB(

26);USING U\$;VENT#(i):NEXT:LPR

INT CHR\$(12)

WINDOW CLOSE 1:GOSUB Me03:

CLOSE:RETURN

entete:

Page=Page+1:Li=0

LPRINT CHR\$(27)CHR\$(81)

LPRINT"Extrait compte banque " LE

FT\$(F\$.LEN(F\$)-2) ", ligne "Npl" A "

Ndl", page "Page

FOR i=0 TO 126:LPRINT "-":NEX

T:LPRINT

LPRINT "I" I" SPC(13) "I"

SPC(33) "I": FOR i=1 TO 5:LPR

INT SPC(12)"I":NEXT:LPRINT "I"

LPRINT "I" DATE ! NUMERO

! LIBELLE ! SPC(23):I RECETT

E ! DEPENSE ! SOLDE

! MOUVEMENT ! SOLDE !P!

LPRINT "I" ! OPERATION

I" SPC(33)"I" BANQUE ! BAN

QUE ! BANQUE ! CARTE BL

EUEI CARTE BLEUEI !

FOR i=0 TO 126:LPRINT "-":NEX

T:LPRINT

RETURN

Totaux:

FOR i=0 TO 126:LPRINT "-":NEX

T:LPRINT

LPRINT "I" Totaux":LPRINT TAB(60)

"I":LPRINT USING U\$;TD#::LPR

INT "I":LPRINT USING U\$;TC#::LP

RINT "I":LPRINT USING U\$;So#::

LPRINT "I" USING U\$;TCB#::LPR

INT "I":LPRINT USING U\$;CB#::LP

RINT "I" !

FOR i=0 TO 126:LPRINT "-":NEX

T:LPRINT

RETURN

Ligne:

LPRINT "I" LEFT\$(zDA\$,2)"/" MID\$(

zDA\$,3,2)"/" RIGHT\$(zDA\$,2) "I" zN

um\$;TAB(26):I"zLIB1\$;TAB(60):"I":

IF Co=1 THEN TD#=TD#+MT#:LPR

INT USING U\$;MT#;

LPRINT TAB(73):"I":

IF Co=2 OR Co=4 THEN LPRINT U

SING U\$;MT#;

LPRINT TAB(86) "I":LPRINT USIN

G U\$;So#::LPRINT TAB(99) "I":

IF Co=3 THEN TCB#=TCB#+MT#:LPR

INT USING U\$;MT#;

IF Co=4 THEN LPRINT USING U\$

;MT#;

LPRINT TAB(112) "I":LPRINT USI

NG U\$;CB#::LPRINT "I":

IF CVI(zPoin\$) THEN LPRINT"";

LPRINT TAB(127) "I"

IF Co=2 THEN TC#=TC#+MT#

IF Co=4 THEN TC#=TC#+MT#:TCB#=T

CB#+MT#

IF Co=2 OR Co=3 THEN VENT#(CV

I(zVent\$))=VENT#(CVI(zVent\$))+MT#

IF CVI(zPoin\$) THEN suite

IF Co=1 THEN SBT#=SBT#+MT#

IF Co=2 THEN SBT#=SBT#+MT#

IF Co=3 THEN SCT#=SCT#+MT#

IF Co=4 THEN SBT#=SBT#+MT#:SCT#

=SCT#+MT#

suite:

RETURN

AffCons:

GET 1,NumF:MOVETO 80,85:PRINT

LEFT\$(zDA\$,2)"/" MID\$(zDA\$,3,2)

+ "/" + RIGHT\$(zDA\$,2)

MOVETO 232,85:PRINT zNum\$:MO

VETO 80,109:PRINT zLib1\$:MOVE

TO 80,133:PRINT zLib2\$:MOVETO

80,157:PRINT USING U\$;CVD(zM

on\$)

MOVETO 220,229:PRINT USING U

\$;CVD(zSol\$):MOVETO 220,253:PR

INT USING U\$;CVD(zCB\$)

code=CVI(zCode\$):BUTTON code+11,

2:IF CVI(zPoin\$) THEN BUTTON

11,2

IF CVI(zVent\$) THEN BUTTON CV

I(zVent\$),2

RETURN

Allô ProDOS ? :

Troisième volet de notre découverte de ProDOS (voir "ProDOS à l'essai" -Pom's 12- et "Pot-pourri ProDOS" -Pom's 16-), nous allons aborder la programmation en assembleur sous ProDOS. Nous verrons les protocoles de co-résidence de plusieurs programmes et l'adjonction de nouvelles commandes à ProDOS avec deux applications pratiques : un TDUMP de visualisation du contenu des fichiers, et un INIT de formatage des disquettes évitant d'utiliser l'utilitaire Filer.

Sous DOS 3.3, Apple n'avait que très peu documenté les conventions d'appel en assembleur : les méthodes d'accès au File Manager n'étaient même pas effleurées par le manuel du DOS ! Résultats : les nombreux aventuriers qui s'étaient hasardés à explorer le système avaient chacun emprunté une approche différente, et le DOS s'est vite retrouvé truffé de "patches" et de modifications internes. Une telle situation a vite

découragé les programmeurs de Cupertino d'envisager le développement d'un DOS 3.4 car celui-ci, ayant nécessité un ré-assemblage et donc la modification de la plupart des adresses des routines, aurait rendu incompatible un très grand nombre de programmes.

Fini le "bidouillage" !

Une philosophie différente existe pour ProDOS : le "Technical Reference Manual" décrit en détail les points d'entrée de ProDOS et du Basic.System mais repose sur un double engagement : celui des programmeurs de n'utiliser que les points d'entrée officiels, de ne pas appeler directement les routines internes, ni occuper les places libres en mémoire, celui d'Apple de conserver la compatibilité de ses points d'entrée avec les versions futures de ProDOS.

Plusieurs versions de ProDOS ont déjà été commercialisées : 1.0

en janvier 1984, 1.0.1 en juillet 1984; on parle d'une version prochaine gérant le futur modem d'Apple... Attention: la version "B.4" que possède certains lecteurs de Pom's, distribuée aux développeurs de logiciels à partir de janvier 1983, n'est pas officiellement supportée par Apple (les fichiers ProDOS et Basic.System, appelés Pro. Kernel et Bas.Interp, ont depuis été corrigés et réassemblés).

Gestion de la mémoire

Jusqu'à présent, les programmes assembleurs étaient généralement placés en page 3 (\$300-\$3CF) ou juste au-dessous des trois buffers du DOS en \$9600. D'où une joyeuse cacophonie lorsqu'il s'agissait de faire résider plusieurs d'entre eux en mémoire, ou d'utiliser simultanément plus de trois fichiers !

Cette anarchie est révolue sous ProDOS : les programmes devront pouvoir co-exister en

Pgm INIT.CODE0.S (Assembleur Big Mac)

```

1      LST OFF
2 *
3
4 *****
5 *
6 *          PRODOS INIT
7 *
8 *****
9
10 *****
11 * Copyright (C) 1985 Alexandre Avrane *
12 *
13 * Modifié: 27/04/85
14 * Créé: 10/03/85
15 *
16 * Assembleur: Big Mac
17 *****
18
19
20 * Objectifs:
21 *
22 * Initialisation d'une disquette ProDOS
23 * sans utiliser le Filer
24 *
25 * Syntaxe:

```

```

25 *
26 * INIT /<nom volume> [.S <slot>]
27 *      [,D <drive>]
28 *
29 * Environnement: ProDOS + Basic System
30 * Compatibilité: ProDOS B.4, 1.0, 1.0.1 et
31 * suivants
32 * Intégration: commande externe Basic
33 * System
34 *
35 * Organisation interne:
36 *
37 * 1- Initialiseur (géré par CMDLOAD)
38 * -cherche place disponible en RAM
39 * -relogue les modules suivants
40 * -chaîne ode externe à ProDOS
41 * 2- Contrôleur de commande
42 * -vérifie la syntaxe de la commande
43 * -sinon exc ou envoi vers autre ode
44 * 3- Formateur physique
45 * -formate pistes en 16 secteurs
46 * -contrôle le formatage
47 * 4- Formateur logique
48 * -initialise les blocs au format
49 * logique ProDOS
50 *
51 * Link
52 *
53 * Ce source d'assemblage doit être

```

```

54 * chaîné avec le fichier
55 * des blocs du boot ProDOS (correspondant
56 * à la version
57 * désirée) par le programme INIT.LINK,
58 * puis avec CMDLINK.
59 *
60 * Source (Big Mac): INIT.CODE0.S
61 * Objet:
62 * Link:
63 * Module exécutable: INIT.CODE0
64 *
65 * Constantes d'assemblage
66 *
67 * 60 GAP1 = 255 début de piste
68 * 61 GAP2 = 5 entre champs
69 * adresse à données
70 * (5-10)
71 * 62 GAP3 = 75 entre secteurs
72 * (16-28) par défaut
73 * 63 RETRY = 16 nb essais pour
74 * une piste
75 * 64 B = $20 caractère blanc
76 *
77 * Adresses utilisées:
78 *
79 * 68 PTR = $40 vecteur temporaire
80 * 69 RINEM = $73
81 * 70
82 * 71 PAINTER = $B00C affiche erreur

```


appels en assembleur

mémoire, quel que soit le nombre de fichiers ouverts. A cet effet, ils disposent d'une carte des pages RAM libres (située en \$BF58-\$BF6F), et doivent suivre le protocole suivant :

- les programmes courts et temporaires (chargés puis exécutés qu'une seule fois) se placent en \$300 ;
- les programmes temporaires ne faisant pas d'appels à ProDOS doivent examiner la carte de la mémoire et se placer sur les plus hautes pages disponibles ;
- en règle générale, les programmes devront également marquer sur cette carte les pages 6502 qu'ils s'allouent, afin de ne pas être écrasés par les programmes suivants ou par ProDOS lorsqu'il gère ses buffers.

La carte de la mémoire contient 24 octets, soit 192 bits ; chacun représente le status d'une des pages \$00 à \$BF, le bit est à 1 si la page est libre. Les pages \$00 à \$02 sont toujours marquées occupées, ainsi que les pages \$9A à

\$BF lorsque le Basic System est présent.

A l'origine, une routine GETBUFR, située en \$BEF5, devait pouvoir être appelée pour obtenir le numéro de la première page disponible. Celle-ci a disparue des versions actuelles de ProDOS, probablement pour des raisons de place.

Heureusement, son fonctionnement a pu être reconstitué et vous la trouverez au sein du programme CMDLOAD qui sera détaillé par la suite.

En entrée de cette routine, l'accumulateur doit contenir le nombre de pages de \$100 octets désirés et le registre Y doit être non nul si le programme est permanent et doit donc marquer les pages comme définitivement occupées.

En sortie, la retenue (carry) est à 1 s'il n'existe pas de groupe de pages consécutives libre ; sinon l'accumulateur contient le numéro de la première page du groupe. Auparavant, nous devons parler protocole :

Mise en place d'un programme assembleur

Plusieurs étapes permettent d'intégrer un programme dans l'environnement existant. Comme elles sont relativement complexes, elles ont été reprises en majorité dans le programme CMDLOAD qui initialise les commandes additionnelles (également appelées externes) de ProDOS :

- 1) se charger en \$2000. Cette adresse est supposée libre car les programmes assembleurs doivent être chargés en tout début, avant l'exécution d'un programme Applesoft par exemple ;
- 2) d'abord vérifier le système d'exploitation: si celui-ci n'est pas ProDOS, autant éviter de poursuivre car on court à la catastrophe ;
- 3) vérifier qu'il n'existe pas de fichiers ouverts car ProDOS gère la carte de la mémoire pour ses

```

72 ATADUR = $BE50 sortie cde extern
73 XLEN = $BE52 long cde externe
74 XCNUM = $BE53 n° de cde ProDOS
75 FBITS = $BE54 bits d'autorisation paramètres
76 FBITS = $BE56 bits des paramètres trouvés
77 BI_SLOT = $BE61 slot commande
78 BI_DRIVE = $BE62 drive d'une cde
79 VPATH1 = $BE6C vecteur nom volum
80 BADCALL = $BE6B convertit erreurs MLI->Basic System
81 MLI = $BF00 entrée MLI vers ProDOS Kernel
82 DATETIME = $BF06 appel à routine de l'horloge
83 SYSERR = $BF0F code erreur MLI
84 DATE = $BF90 date/heure sur 4 octets
85
86 WAIT = $FCAB attente
87 RTS = $FF5B contient un RTS
88
89 * Macro-instructions:
90 *
91 WRITE MAC ;écriture physique sur disk (x = slot x $10)
92 STA SC08D,X
93 CMP SC08C,X
94 <<<

```

```

95 DELAI2 MAC ;délai 2 cycles
96 NOP
97 <<<
98 DELAI3 MAC ;délai 3 cycles
99 JMP #+3
100 <<<
101 DELAI4 MAC ;délai 4 cycles
102 >>> DELAI2
103 >>> DELAI2
104 <<<
105 DELAI6 MAC ;délai 6 cycles
106 >>> DELAI2
107 >>> DELAI2
108 >>> DELAI2
109 <<<
110 DELAI7 MAC ;délai 7 cycles
111 PHA
112 PLA
113 <<<
114 DELAI12 MAC ;délai 12 cycles
115 JSR RTS
116 <<<
117
118 ORG $2100 (CMDLOAD de $2000 à $20FF)
119
120 *****
121 * 1 - INITIALISEUR *
122 *****
123

```

```

124 * Doit débiter sur une frontière de page 6502
125
126 * 2.1 - vérifie que la commande externe est la nôtre
127 *
128 START = *
129 CLD ;nécessaire
130
131 * Conventions de link avec CMDLOAD:
132 LDA #>FIN+$100
133 LDA #>LONG
134 V_OLDCHD LDA RTS
135
136 * Conventions de link avec INIT.LINK:
137 VC_BOOT2 LDA BOOT_BL2 convention pour link avec blocs 0/1 ProDOS
138
139
140 *****
141 * 2 - CONTROLEUR DE COMMANDE *
142 *****
143
144 LDA VPATH1 pointeur vers la commande actuelle
145
146 STA PTR
147 LDA VPATH1+1
148 STA PTR+1
149 * (si lecture en $200, on devrait éviter les caractères blancs)

```


propres besoins en buffer. Si, d'autre part, un programme Applesoft a déjà créé des variables chaînes de caractères, celles-ci seront inutilisables et il est préférable d'effectuer un CLEAR après l'initialisation ;

4) appeler la routine ci-dessus de réservation en mémoire,

5) se reloger sur les pages allouées précédemment. Si votre programme est directement relogeable (pas de JSR, JMP, LDA ou STX internes par exemple), il vous suffit d'appeler la routine MOVE du moniteur. Sinon, pas de problème, une routine de CMDLOAD vous mûchera le travail en respectant le protocole suivant :

- la partie à reloger débute en \$2100
- l'octet haut de l'adresse de fin de votre programme, incrémenté de 1, se trouve en \$2102
- l'octet haut de la longueur de votre programme se trouve en \$2104
- l'adresse de la commande externe ProDOS précédente est en \$2106.\$2107. Votre programme devra y aller si la commande ProDOS n'est pas pour lui
- la partie codes instructions 6502 est séparée de la partie

données (constantes + variables) par un BRK (\$00). La partie code sera ré-assemblée, la partie données sera simplement recopiée.

Attention : des instructions du type :

```
PARTIE1 LDA #<PARTIE2 (avec
PARTIE2 quelque
part dans votre
programme)
LDY #>PARTIE2
```

sont à banir car non relogeables, et devront être remplacées par :

```
PARTIE1 LDA PARTIE2
LDA PARTIE1+1
LDY PARTIE1+2
```

Une sixième étape est nécessaire si vous souhaitez raccorder votre programme à ProDOS comme une commande externe: d'abord sauvegarder l'adresse de la commande externe précédente (en \$BE06.BE07), puis placer la vôtre à la place. Bien sûr, la même chose doit s'appliquer si vous vous raccordez ailleurs, à l'Ampersand (&) par exemple : vous devez toujours supposer que vous êtes le deuxième programme à vous connecter, et sauver l'ancien vecteur en conséquence.

Nous supposerons désormais que votre programme est une

commande externe, c'est à dire directement accessible au clavier en mode direct, ou par PRINT CHR\$(4) en mode différé dans un programme.

Utilisation de CMDLOAD

Si la lecture des derniers paragraphes vous a rendu tout pâle, et que vous vous dites que, tout compte fait, l'empirisme du DOS 3.3 était bien agréable, dites vous que la localisation absolue en mémoire des programmes est très égoïste et peu réaliste avec la venue prochaine de processeurs 16 bits sur Apple II. De plus, le module CMDLOAD vous fait tout le travail !

CMDLOAD est destiné à être utilisé par tout programme désireux de se connecter à ProDOS. Les deux exemples ci-dessous (TDUMP et INIT) ne sont destinés qu'à illustrer son utilisation.

Chargez par BLOAD d'abord votre programme en \$2100 puis exécutez CMDLOAD en \$2000, ou utilisez le programme CMDLINK qui fusionne les deux en un fichier ProDOS de type binaire, directement exécutable par "-" (smart Run), par exemple

```
149
150 LDY #1
151 COMPARE LDA (PTR),Y obtient un caract
152 CMP COMMANU-1,Y c'est "INIT" ?
153 BNE NO_CMD non
154 INY
155 CPY #4+1
156 BCC COMPARE
157
158 * 2.2 - Demande à ProDOS d'examiner la
suite de la commande
159 * -----
160 DEY
161 DEY
162 STY XLEN stocke long - 1
163 LDA #0
164 STA SYSERR initialise code
erreur MLT
165 STA XCNUM indique cde exter
166 LDA #00000001
167 STA PBITS nom de fichier
(volume) autorisé
168 LDA #00000100
169 STA PBITS+1 slot & drive
autorisés
170
171 LDA PRINTERR+1 détourne le
vecteur erreur
(pas de préfixe)
172 STA SAVE_ERR
173 LDA PRINTERR+2
174 STA SAVE_ERR+1
175 V_SUITE LDA SUITE vecteur de retour
après examen
176 LDA V_SUITE+1
```

```
177 STA XTADDR indique à ProDOS
où revenir après
178 STA PRINTERR+1 indique où
revenir si erreur
179
180 LDA V_SUITE+2
181 STA XTADDR+1
182 STA PRINTERR+2
183 CLC ;indique à ProDOS
que c'était pour
nous
184 RTS
185 NO_CMD SEC ;code pas pour
nous
186 JMP (V_OLDCMD+1) alimenté par
CMDLOAD
187
188 * 2.3 - Retour d'analyse de la commande
189 * -----
190 SUITE = *
191 LDX SAVE_ERR remplace le vecteur
d'erreur
192 STX PRINTERR+1
193 LDX SAVE_ERR+1
194 STX PRINTERR+2
195
196 CMP #0 si pas d'erreur,
on re-formatte
197 BEQ SUITE_2
198 CMP #6 path not found
autorisé
199 BEQ SUITE_2
200 CMP #8 i/o error
autorisé
201 BNE GO_BAD2
```

```
202 SUITE_2 LDA RI_SLOT slot demandé
203 ORA #800
204 STA V_CARD+2
205 LDA #628 no devlow
206 V_CARD LDX #C0FF si pas un Disk II
207 BEQ LIT_VOL
208
209 GO_BAD JSR BADCALL en entrée: Acc-
code erreur MLT
210 GO_BAD2 JMP PRINTERR
211
212 * 2.4 - Extraction du nom du volume
213 * -----
214 LIT_VOL = * (PTR pointe déjà
vers VFATH1)
215 LDY #0
216 LDA (PTR),Y longueur du nom
du volume
217
218 TAX
219 DEX
220 CPX #810
221 BCS SYNTAX nom de volume <
16 caractères
222 STX NOM_LG
223 INY
224 LDA (PTR),Y
225 CMP # '/'
226 BNE SYNTAX le 1er caractère
doit être "/"
227 INY
228 LDA (PTR),Y le 1er caractère
doit être une
lettre
229 LIT_CAR CMP # 'A'
230 BCC SYNTAX
```


à partir de votre programme de STARTUP.

Dans ce dernier cas, votre programme doit s'appeler xxx.CODE, et le fichier résultant s'appellera simplement xxx. (Apple a créé sous ProDOS des fichiers de type "commande externe", numérotés \$F0, mais ceux-ci ne sont pas exécutables par BRUN ou "-"... quel intérêt ?)

CMDLOAD peut générer les messages d'erreur :

FILE(S) STILL OPEN (aucun fichier ne doit être ouvert pour la mise en place d'une commande) ;
PROGRAM TOO LARGE (s'il ne reste pas suffisamment de pages libres en mémoire pour reloger la commande).

Fonctionnement des commandes externes

Lorsque ProDOS rencontre une commande qu'il ne connaît pas, il appelle l'adresse EXTRNCMD (\$BE06). En temps normal, celle-ci contient un JMP vers une instruction RTS. Mais il pointe sur le début de votre programme (\$2100) si celui-ci a été initialisé

par CMDLOAD.

Votre programme doit vérifier qu'il s'agit bien d'une commande qui lui est destinée. La chaîne de caractères a été placée par ProDOS dans le buffer d'entrée en \$0200 (avec éventuellement les caractères blancs non significatifs qui ont pu être saisis), et ses huit premiers octets (sans caractère blanc) se trouvent dans le buffer dont l'adresse est pointée par VPATH1 (\$BE6C-BE6D). L'oubli de ce contrôle détruirait la chaîne d'interception et, par exemple, aucune instruction Applesoft ne serait disponible en mode direct...

S'il ne s'agit pas de votre commande, effectuez un saut à l'adresse contenue à l'adresse (relogée) \$2106.2107 avec la retenue (carry) à 1.

Extraction des paramètres par ProDOS

Si votre commande ne nécessite aucun paramètre, ou si vous préférez les extraire vous même, placez un \$00 en PBITS et PBITS+1 (\$BE54-BE55) et retournez à ProDOS par un RTS avec la retenue à zéro.

Dans le cas contraire, il faut indiquer à ProDOS quels sont les paramètres autorisés en alimentant les bits de PBITS et PBITS+1 :

\$8000 demande de concaténer le préfixe actuel avec le nom de fichier saisi
\$4000 seul un numéro de slot est autorisé (PR# ou IN#)
\$2000 commande en mode différé uniquement (OPEN READ...)
\$1000 nom de fichier autorisé
\$0800 création automatique du fichier saisi s'il n'existe pas
\$0400 paramètre T autorisé
\$0200 second nom de fichier obligatoire (RENAME)
\$0100 nom de fichier obligatoire
\$0080 à \$0001 autorise respectivement les paramètres A, B, E, L, @, S/D, F et R

Il faut également indiquer à ProDOS que vous êtes une commande externe en stockant un \$00 en XCNUM (\$BE53), l'adresse où ProDOS devra vous rappeler après l'analyse des paramètres en XTRNADDR (\$BE50-BE51), la longueur diminuée de 1 du nom de votre commande en XLEN (\$BE52) (par exemple 3 pour la commande

```

230      CMP  #'A'+1
231      BCS  SYNTAX
232 LIT_NOM STA  NOM_VOL-2,Y
233      DEX
234      BEQ  GO_GO      fin du nom
235      INY
236      LDA  (PTR),Y    caractère suivant
237      CMP  #'.'        "." est autorisé
238      BEQ  LIT_NOM
239      CMP  #'0'
240      BCC  LIT_CAR
241      CMP  #'9'+1
242      BCC  LIT_NOM    c'est un chiffre
243      BCS  LIT_CAR
244 SYNTAX  LDA  #$40
245      BNE  GO_BAD    =jmp
246
247 * 2.5 - Formattages
248 *
249 GO_GO   JSR  FORMAT_P    format. physique
250      BCS  GO_BAD
251      JSR  FORMAT_L    format. logique
252      BCS  GO_BAD
253      RTS
254
255
256 *****
257 *      3 - FORMATEUR PHYSIQUE      *
258 *****
259
260 * 3.1 - Sélection slot & drive
261 *
262 FORMAT_P =
263      LDA  BI_SLOT    no slot: $0a
264      ASL

```

```

265      ASL
266      ASL
267      ASL
268      STA  F$LOT      $a0
269      ORA  BI_DRIVE    $000000dd
                        (drive = 1 ou 2)
270      TAX
271      DEX
272      LDA  $C09A,X    sélectionne drive
                        (X = $0555000d)
273
274 * 3.2 - Recalibre la tête de lecture
275 *
276      JSR  RECALIBR
277      LDA  $C0B0,X    désactive toutes
                        les phases
278      LDA  $C0B2,X
279      LDA  $C0B4,X
280      LDA  $C0B6,X
281
282 * 3.3 - Boucle sur les 35 pistes
283 *
284      LDA  $GAP3      espacement
                        inter-secteurs
                        par défaut
285      STA  GAP3_BIS
286      LDA  #$80
287      STA  OP_MLI      on appellera
                        ProDOS en lecture
288 LOOPTRK LDA  $RETRY    nombre d'essais
                        avant abandon
289      STA  RETRYCNT
290      INC  GAP3_BIS    vitesse lecteur
291 INITHORE LDA  $0      on commence par
                        le secteur 0

```

```

292      STA  F$ECTOR
293      SRT
294      JSR  INTRACK    format. physique
295      BCS  ENDINIT    write-protect
296
297      JSR  VER_TRK    vérifie lg piste
298      CLI
299      BCC  GOCHECK    ok longueur
300      DEC  GAP3_BIS    on diminue GAP3
                        et on recommence
301      LDA  GAP3_BIS
302      CMP  $8
                        abandon si - de 8
                        oct intersecteurs
303      BCS  INITHORE
304 IO_ERR  LDA  $27      i/o error
305      SEC
306      BCS  ENDINIT    =jmp
307
308 GOCHECK JSR  CHECKTRK    vérifie le
                        formatage
309      BCC  NEXTTRK
310      DEC  RETRYCNT
311      BNE  INITHORE    on refait un
                        essai
312      BEQ  IO_ERR      =jmp
313
314 NEXTTRK LDA  F$TRACK
315      INC  F$TRACK
316      LDY  F$TRACK
317      JSR  ARM_MOVE    place bras lecteur
318      CMP  $35          35e piste ?
319      BCC  LOOPTRK     pas encore
320      CLC
321 ENDINIT LDY  $C0B8,X    arrête le moteur
322 EXIT   RTS

```


INIT). Enfin, retournez à ProDOS par un RTS avec la retenue à zéro.

ProDOS se chargera d'examiner les paramètres, et de générer éventuellement les messages d'erreur de SYNTAX, RANGE ou PATH NOT FOUND (si des paramètres slot et drive ou un nom complet de fichier avec l'indication du volume ont été saisis, ProDOS lit les disques en ligne pour créer ses pointeurs internes entre volume logique et lecteur physique).

Si tout se passe bien, ProDOS rappelle votre programme à partir de XTRNADDR. Les paramètres effectivement trouvés sont indiqués par un bit de présence en FBITS et FBITS+1 (\$BE56-BE57), qui reprend le découpage de PBITS. La valeur effective des paramètres est stockée respectivement en :

VADDR \$BE58-BE59 paramètre A
VBYTE \$BE5A-BE5C paramètre B
VENDA \$BE5D-BE5E paramètre E
VLNTH \$BE5F-BE60 paramètre L
VSLT \$BE61 paramètre S
VDRIV \$BE62 paramètre D
VFELD \$BE63-BE64 paramètre F
VRECD \$BE65-BE66 paramètre R
VVOLM \$BE67 paramètre V
(ignoré par les commandes ProDOS actuelles)

VLIN \$BE68-BE69 paramètre @
RVTYPE \$BE6A paramètre T
VIOSLT \$BE58-BE59 slot PR#/IN#
VPATH1 \$BE6C-BE6D vecteur vers nom fichier
VPATH2 \$BE6E-BE6F vecteur vers second fichier

Ces deux derniers vecteurs pointent sur un buffer qui contient d'abord la longueur du nom sur un octet, puis les caractères du nom (bit fort à zéro). Si le préfixe a été demandé (premier bit de PBITS), le chemin complet d'accès au fichier y est stocké.

Votre programme est maintenant libre de poursuivre son exécution.

Appels à ProDOS en assembleur

Sous DOS 3.3, il était possible d'envoyer des commandes en les préfixant par un CTRL-D et en les envoyant à la routine de sortie des caractères (COUT en \$FDED).

Sous ProDOS, en revanche, il faut stocker la chaîne de caractères dans le buffer d'entrée en \$0200 (avec les bits forts à 1), la faire suivre par un retour-chariot (\$8D) puis appeler

DOSCMD en \$BE03. La petite routine ci-dessous fournit en exemple le catalogue du préfixe courant :

```
LDY #0
LOOP LDA CMD,Y
STA $200,Y
INY
CMP #S8D
BNE LOOP
JMP $BE03
CMD ASC "CAT"
DFB $8D
```

Cette méthode ne fonctionne bien sûr que si le Basic.System est actif. On peut heureusement appeler directement le noyau de ProDOS, avec la syntaxe générale :

JSR \$BF00
DFB FONC ; fonction demandée
DA PARM ; adresse table paramètres
BCS ERROR ; suite exécution

En retour, la retenue est à 1 si une erreur s'est produite et l'accumulateur contient la valeur du code erreur.

Les tables de paramètres sont différentes pour chaque fonction et débutent toujours par un octet indiquant le nombre en hexadécimal de paramètres suivants ; voici leurs fonctions

```
323 * (en sortie: 0-0 -> formatage ok,
324 *      0-1 A-27 -> i/o error,
325 *      A-2B -> write protect)
326
327 * 3.4 - Ecriture de l'en-tête de piste
328 * -----
329 INTRACK = "
330 LDY #5          initialise la
                      vérif de lg
331 LDA FTRACK
332 PHA
333 JSR VER_4_4
334 PLA
335 EOR #1
336 LDY #9
337 JSR VER_4_4
338
339 LDA $C089,X      moteur en marche
340
341 SEC
342 LDA #S2B
343 LDY $C08D,X      mode écriture
344 LDY $C08E,X
345 *
346 * A partir d'ici chaque cycle-machine
    compte:
347 * 4 cycles-machine sont nécessaires pour
    écrire 1 bit.
348 * 40 cycles x (GAP1 + 16 x (GAP2 +
    GAP3))
349 * + 32 cycles x 16 x (14 + S1S7 + T)
350 * (Les branchements relatifs doivent rester
    sur la me "me page)
351 *
352 BMT EXIT         disq protégée
```

```
353 >>> DELAT7
354 LDA #SFF
355 STA $C08F,X      d'abord un octet
                      à SFF sur 40 cycl
356
357 CMP $C08C,X
358 LDY #GAP1
359 >>> DELAT4
360 >>> DELAT4
361 JSR GAPWRITE      puis plusieurs
                      autres pour
                      synchroniser
362 * 3.5 - Ecriture de la zone adresse
363 * -----
364 LOOPSECT LDA #SD5      écrit D5 AA 96
                      sur 32 cycles
365 JSR CONV_4E3
366 LDA #SAA
367 JSR CONV_4E3
368 LDA #S96
369 JSR CONV_4E3
370
371 LDA #1            # de vol toujours
                      1 sous ProDOS
372
373 NOP
374 JSR CONV_4_4
375 LDA FTRACK        # de piste
376 JSR CONV_4_4
377 LDA FSECTOR        # de secteur
378 JSR CONV_4_4
379
380 LDA FTRACK        calcul du contrôl
                      de somme
381
382 EOR FSECTOR
383 EOR #1
384 PHA
```

```
383 LSR
384 ORA #SAA          on le transforme
                      en nibble valide
                      et on l'écrit
385 >>> WRITE
386 PLA
387 ORA #SAA
388 JSR CONV_4E3      2e nibble du
                      contrôle de somme
389
390 LDA #SDE          écrit DE AA EB
391 JSR CONV_4E3
392 LDA #SAA
393 JSR CONV_4E3
394 LDA #SEB
395 JSR CONV_4E3
396
397 * 3.6 - Ecriture de la zone de données
398 * -----
399 LDY #GAP2          octets entre
                      adress et données
400 JSR GAP_L3
401
402 LDA #SD5          puis D5 AA AD
403 JSR CONV_4E3
404 LDA #SAA
405 JSR CONV_4E3
406 LDA #SAD
407 JSR CONV_4E3
408
409 LDY #S56          on écrit S56
                      nibbles sur 32
                      cycles
410 BNE DATA_L3      =jmp
411 DATA_L2 >>> DELAT6
412 DATA_L3 LDA #S96      S96 - S00 après
```


(reportez vous à la bibliographie pour en connaître la description, ainsi que les codes erreurs) :

Voir liste 1 ⇒

Voici, à titre d'exemple, une routine fournissant les attributs du fichier "ProDOS" sur le préfixe courant :

Voir liste 2 ⇒

Le principal utilisateur de ces appels au noyau de ProDOS est le Basic.System ; toutes les fonctions d'accès aux fichiers sont officiellement disponibles via la page globale \$BE. Voilà qui simplifie la vie, car la routine ci-dessus devient :

Voir liste 3 ⇒

ProDOS étant indépendant du type de lecteur de disque employé, les fonctions ci-dessus parviennent finalement à un Device Driver ("gérant d'unité") spécifique. Pour la petite histoire, quatre fonctions sont possibles: STATUS fournit le nombre total de blocs du support et son status de protection contre l'écriture, READ et WRITE n'appellent aucun commentaire, FORMAT n'est malheureusement pas disponible pour le Disk II d'Apple (... sinon pourquoi proposerait-on un utilitaire de

Liste 1

\$40	ALLOC_INTERRUPT	fixe une routine de gestion d'interruption masquable (IRQ)
\$41	DEALLOC_INTERRUPT	retrait d'une routine d'interruption
\$65	QUIT	demande de chargement d'un nouvel interpréteur système
\$80	READ_BLOCK	lecture d'un bloc de 512 octets
\$81	WRITE_BLOCK	écriture d'un bloc de 512 octets
\$82	GET_TIME	appel à l'horloge pour mise à jour date/heure (\$BF90-BF93)
\$C0	CREATE	création d'un nouveau fichier
\$C1	DESTROY	destruction d'un ancien fichier
\$C2	RENAME	renomme un ancien fichier sous le m ^{ême} chemin d'accès
\$C3	SET_FILE_INFO	change les attributs d'un fichier (type, lock/unlock,...)
\$C4	GET_FILE_INFO	lit les attributs courants d'un fichier
\$C5	ONLINE	obtient, pour un lecteur physique, le nom du volume monté
\$C6	SET_PREFIX	change le préfixe par défaut
\$C7	GET_PREFIX	lit le préfixe courant
\$C8	OPEN	ouvre un fichier et retourne un numéro de référence
\$C9	NEWLINE	spécifie le caractère de fin de champ (normalement \$0D)
\$CA	READ	lit un ou plusieurs octets à partir de la position courante
\$CB	WRITE	écrit un ou plusieurs octets
\$CC	CLOSE	ferme un, plusieurs ou tous les fichiers ouverts
\$CD	FLUSH	écrit sur disque les buffers de un ou plusieurs fichiers
\$CE	SET_MARK	change la position courante de lecture/écriture
\$CF	GET_MARK	obtient la position courante de lecture/écriture
\$D0	SET_EOF	change la position de fin de fichier (permet de tronquer)
\$D1	GET_EOF	obtient la position de fin de fichier
\$D2	SET_BUF	change l'adresse du buffer d'un fichier
\$D3	GET_BUF	obtient l'adresse du buffer d'un fichier

Liste 2

	JSR \$BF00	; appelle ProDOS
	DFB \$C4	; get_file_info
	DA PARM	; table des paramètres
	BCS ERREUR	; erreur ?
	RTS	; non : fini
ERREUR	JSR \$BE8B	; traduit le code erreur ProDOS en code erreur Basic.System
	JMP \$BE0C	; gère l'erreur et quitte
PARM	DFB 10	; 10 paramètres suivent...
	DA PATHNM	; en entrée le nom du fichier, maintenant en sortie :
	DFB 0	; bits d'accès en lecture/écriture/destroy/rename/sauvegarde
	DFB 0	; type de fichier (programme Applesoft, binaire, texte, etc.)
	DA \$0000	; type auxiliaire (par exemple adresse de chargement)
	DFB 0	; type de stockage (sous-catalogue ou niveau des blocs d'index)
	DA \$0000	; nombre de blocs utilisés
	DA \$0000	; date de dernière modification (bits = AAAAAAMMMMMJJJJJ)
	DA \$0000	; heure de dernière modification (bits = HHHHHHHHMMMMMMMM)
	DA \$0000	; date de création
	DA \$0000	; heure de création
PATHNM	DFB \$06	; longueur du nom
	ASC 'PRODOS'	; avec bits forts à zéro

Liste 3

V_PATH	LDA PATHNM	; instruction pour relocation d'adresse
	LDA V_PATH+1	; pointe vers le nom de fichier...
	STA \$BEB4+1	; ...et le stocke sur la table des paramètres (\$BEB4-BEC5)
	LDA V_PATH+2	;
	STA \$BEB4+2	;
	LDA \$C4	; code de get_file_info
	JSR \$BE70	; Basic.System appelle ProDOS avec sa table
	BCS ERROR	; erreur ?
	RTS	; non : fini, les paramètres retournés sont en \$BEB7-BEC5
ERROR	JMP \$BE09	; Basic.System va gérer l'erreur
PATHNM	DFB \$06	; longueur du nom
	ASC 'PRODOS'	; avec bits forts à zéro

413	>>> DELA16	trans-codage
414	>>> DELA14	
415	>>> WRITE	
416	DEY	
417	BNE DATA_L2	
418	DEQ DATA_L5	-jmp
419		
420 DATA_L4	>>> DELA12	puis les \$100
		octets suivants
421 DATA_L6	>>> DELA12	total: \$156, -
		\$100 octets après
		codage
422	>>> DELA14	
423	>>> WRITE	
424	DEY	
425	BNE DATA_L4	
426		
427	LDA \$596	contrôle de somme
		des données
428	>>> DELA12	
429	JSR CONV_4E3	
430		
431	LDA \$0DE	écrit DE AA EB FF
432	JSR CONV_4E3	
433	LDA \$5AA	
434	JSR CONV_4E3	
435	LDA \$5EB	
436	JSR CONV_4E3	
437	LDA \$5FF	
438	JSR CONV_4E3	
439		
440	LDY GAP3_B18	octets sync
		inter-secteurs
441	JSR GAP_L0	
442		

formatage dans cet article ?). De toute manière, il est hautement déconseillé de l'appeler directement car, d'une part les fonctions READ_BLOCK et WRITE_BLOCK de ProDOS suffisent amplement, d'autre part le Device Driver se trouve dans la partie haute de la carte langage (sur l'emplacement du moniteur de la ROM), donc peu accessible et il ne vous pardonnera jamais les erreurs.

Première application : TDUMP

Si on passait aux travaux pratiques ? Pour s'échauffer, abordons la commande TDUMP ; elle fournit sur l'unité courante de sortie (écran vidéo ou imprimante) un "écorché" en trois parties d'un fichier : d'abord la position du dump dans le fichier (décalage par rapport au début), puis l'affichage des codes ASCII correspondants (sauf les codes de contrôle qui font trop souvent délirer les cartes d'interface...), enfin la valeur en hexadécimal de chaque octet.

La syntaxe complète est :

TDUMP <nom fichier> [,S <slot>] [,D <drive>]

Cette commande est applicable aux fichiers de tous types, (y compris les types SYS) ; les fichiers texte à accès direct sont également visualisables en totalité. Dans un but pédagogique, on peut également accéder aux sous-catalogues, ainsi qu'au catalogue principal du volume (par exemple en tapant TDUMP/EXAMPLES).

On peut arrêter momentanément l'affichage par une touche quelconque, puis définitivement par Ctrl-Q.

Afin d'améliorer l'ergonomie, la commande adapte automatiquement l'affichage sur 40 ou 80 colonnes selon l'adresse de l'unité de sortie (si celle-ci pointe sur un slot, l'affichage ou l'impression passe sur 80 colonnes).

Notre commande, appelée par ProDOS (et initialement mise en place par CMDLOAD), doit donc d'abord demander à contrôler la saisie du nom d'un fichier et des paramètres S et D éventuels, puis rendre la main. En retour, il faut ouvrir le fichier, en utilisant le buffer brouillon de ProDOS (utilisé par les commandes

temporaires telles que CATALOG et pointé par HIMEM), boucler sur une routine qui lit 8 ou 16 octets à la fois (selon le nombre de colonnes), puis formater une ligne qui est envoyée à l'affichage. Finalement, n'oublions pas de fermer le fichier !

Le source (Big Mac) s'appelle TDUMP.CODE.S et génère l'objet TDUMP.CODE.

Vous pourrez alors, pour mettre la commande en place, soit taper :
BLOAD TDUMP.CODE
BRUN CMDLOAD
soit exécuter le programme CMDLINK qui créera le fichier TDUMP.

Initialisation d'une disquette ProDOS

Maintenant que nous nous sommes mis à table, passons au plat de résistance : l'initialisation d'une disquette ProDOS.

Certes, l'utilitaire Filer fourni par Apple effectue le même travail. Mais :

1) il interdit de pouvoir initialiser une disquette pendant l'exécution d'un programme puisqu'il faut charger le Filer,

```

443 * 3.7 - Boucle sur secteur suivant
444 * -----
445             INC FSECTOR      avec ProDOS les
                                secteurs sont
                                placés
446             >>> DELAI4      séquentiellement
                                sur la piste
447             >>> DELAI7
448             >>> WRITE      un dernier octet
                                sur 40 cycles
449             >>> DELAI3
450             LDA FSECTOR
451             CMP #810
452             BGE FINSECT      c'est fini pour
                                cette piste
453             JMP LOOPSECT     sinon on
                                repart...
454
455 FINSECT >>> DELAI7
456             >>> DELAI7
457             LDA $C08E,X      remet en mode
                                lecture
458             LDA $C08C,X
459             CLC
460             RTS
461
462 * 3.8 - Module déplacement bras de lecture
463 * -----
464 RECALIBR LDA FSIOT
465             LDA $C089,X      moteur en marche
466             LDA $C08E,X      mode lecture
467             LDA $C08C,X
468
469             LDA #48          prétend être sur
                                la piste 48
470             LDY #0           on veut aller en
                                piste 0

```

```

471
472 ARM_MOVE = *
473 * (en entrée: A=piste actuelle, Y=piste
    désirée)
474 * (en sortie: FTRACK est mis à jour)
475             STA A_TRACK
476             STY D_TRACK
477             LDA #0
478             STA ARM_FLAG
479             LDA A_TRACK
480             SEC
481             SBC D_TRACK      calcule nb de
                                pistes à déplacer
482             BEQ ARM_L4      c'est fini
483             BCS ARM_L2      un continue
484             EOR $FF
485             ADC #1          ajuste résultat
                                si sens négatif
486 ARM_L2     STA ARM_DIFF
487             ROL ARM_FLAG
488             LSR A_TRACK
489             ROL ARM_FLAG
490             ASL ARM_FLAG
491             LDY ARM_FLAG
492 ARM_L3     LDA ARM_TAB,Y
493             JSR PHASE        phase à activer
494             LDA ARM_TAB+1,Y  prend la phase
                                suivante
495             JSR PHASE
496             TYA
497             EOR #2
498             TAY
499             DEC ARM_DIFF
500             LDA ARM_DIFF
501             BNE ARM_L3
502 ARM_L4     LDA D_TRACK

```

```

503             STA FTRACK
504             LDY FSIOT
505             RTS
506
507 PHASE     ORA FSIOT          S=0
508             TAX
509             LDA $C081,X      active la phase
510             LDA #57
511             JSR WAIT         attend 20
                                millisecondes
512             LDA $C080,X      désactive la
                                phase
513             RTS
514
515 * 3.b - conversion octet <-> nibble 4+4
516 * -----
517 CONV_4_4  PHA
518             LSR
519             ORA #9AA         1alc1elg
520             >>> WRITE
521             PLA
522             >>> DELAI6
523             ORA #9AA         1bld1d1fh
524 CONV_4E2  >>> DELAI2
525 CONV_4E3  >>> DELAI2
526             >>> DELAI7
527             >>> WRITE
528             RTS
529
530 * 3.c - Module d'écriture des octets
    synchronisés à 40 cycles
531 * -----
532 GAPWRITE = *
533             en entrée y = nb
                                octets à écrire
534             BNE GAP_L2      =jmp
535             BNE GAP_L4      =jmp
536             >>> DELAI2

```


- puis recharger un interpréteur comme le Basic.System ;
- 2) le Filer est très long (45 secondes) ;
 - 3) le Filer ne s'adapte pas à la vitesse des lecteurs de disques : sur les deux lecteurs que je possède, le Filer refuse de formater toute disquette sur l'un d'entre eux, estimant que la vitesse de rotation est trop rapide, alors qu'il est possible d'initialiser sans problème sous DOS 3.3 !

Notre programme sera appelé comme une commande externe ProDOS, donc accessible en mode direct ou différé. Il devra être plus rapide et s'adapter à la vitesse des lecteurs de disques. Seules les disquettes pour Disk II d'Apple pourront être initialisées, pas question d'accéder à un ProFile. La syntaxe est :

INIT / <nom volume> [,S <slot>] [,D <drive>]

Le nom du volume devra être précédé par un caractère "/" et comprendre au plus quinze caractères répondant à la syntaxe des noms de fichiers ProDOS. Si slot et drive ne sont pas indiqués, l'initialisation s'effectuera sur le lecteur du préfixe courant. Les messages d'erreur possibles

sont :

SYNTAX ERROR
RANGE ERROR
WRITE PROTECTED
NO DEVICE CONNECTED (si aucun lecteur Disk II sur le slot/drive indiqué)
I/O ERROR (si la disquette est physiquement endommagée ou la vitesse du lecteur est largement en dehors de la plage de tolérance)

Bien entendu, l'initialisation d'une disquette efface définitivement toutes les données qui pouvaient y être enregistrées. Deux parties très distinctes recouvrent le terme d'initialisation: le formatage physique et le formatage logique.

Formatage physique

Le formatage physique est fort complexe. La description qui suit n'est nullement nécessaire à l'utilisation du programme, mais tente de répondre à la curiosité de nombreux lecteurs.

Une disquette est composée de 35 pistes, et la tête de lecture peut se déplacer individuellement sur chacune. Que la vie serait douce si tous les lecteurs de disque

avaient exactement la même vitesse de rotation, à la micro-seconde près! C'est malheureusement impossible, et des méthodes complexes de lecture/écriture des quelques 50000 bits d'une piste et d'encryptage/déencryptage de ceux-ci en octets ont dû être développées.

A l'époque de la création du Disk II, une capacité de 80ko était généralement disponible sur les disquettes de 5 pouces 1/4 ; l'abandon par Apple de la méthode coûteuse des bits de synchronisation, puis le découpage d'une piste en 16 secteurs à partir du DOS 3.3, ont porté la capacité à 140k, en acceptant un certain nombre de contraintes qui doivent être gérées par le logiciel :

- un octet ne peut être lu ou écrit que s'il respecte plusieurs limitations (bit fort à 1, pas plus de deux bits nuls consécutifs, etc.) ; en conséquence, il faut 342 octets pour en coder 256.
- chaque secteur d'une piste est composé de deux zones : adresse et données ;
- la zone adresse comprend un préfixe (composé des octets

```

536      >>> DELAI12
537 GAP_L2 >>> DELAI13
538 GAP_L3 >>> DELAI12
539      >>> DELAI13
540 GAP_L4 >>> DELAI12
541      LDA #5FF      donne un octet
                        à 10 bits:
                        1111111100

542      >>> WRITE
543      DEY
544      BNE GAP_L1
545      >>> DELAI4
546      RTS
547
548 * 3.d - Module de vérification du
      formatage d'une piste
549 * -----
550 CHECKTRK = *
551      LDA #0
552      STA M_BLOCK+1
553      STA M_BUFFER      lit vers buffer
                        tous usages
                        qui commencent en
                        Himm
554      LDA Himm+1
555      STA M_BUFFER+1
556      LDA VC_PARM+2      détermine l'ad
                        des paramètres
557      STA VC_MLI+1
558      LDY BI_DRIVE      no de lecteur 1/2
559      DEY
560      TYA
561      LSR              ;=> carry=0 ou 1
562      ROR
563      ORA FSLT      donne %dass0000
564      STA M_UNIT
565      LDY #8          8 blocs par piste
566      LDA PTRACK      piste à contrôler

```

```

567      ASL
568      ASL
569      ASL
570      STA M_BLOCK      1er bloc d'une
                        piste = piste x 8
571      BCC CHECK_L2
572      INC M_BLOCK+1      (bloc > 5FF)
573 CHECK_L2 JSR GO_MLI      appel à PRODUS
                        pour lire le bloc
                        via la zone non
                        translatable
                        en sortie, le
                        moteur est arrêté
574      BCS CHECK_L3
575      INC M_BLOCK
576      DEY
577      BNE CHECK_L2      bloc suivant
578 CHECK_L3 RTS          ; c=0 => OK ;
                        ; c=1 => KO
579
580 * 3.e - Vérification de la longueur de la
      piste formatée
581 * -----
582 * (on vient normalement de boucler une
      piste. On doit
583 * donc revenir sur la zone adresse du
      secteur 0.)
584 VER_TRK LDY #1
585 VER_L1 LDA $C08C,X      lit un octet
586      CMP #5FF
587      BNE VER_L1      on attend d'être
                        synchronisé
588      DEY
589      BPL VER_L1      on laisse passer
                        2 oct synchro
590 VER_L2 LDA $C08C,X
591      RPL VER_L2
592      CMP #5FF
593      BEQ VER_L2      encore un

```

```

synchronisé à laisser passer
594      LNY
595      CFI #13          13 octets en zone
                        adresse
596      BEQ VER_L3      c'est fini
597      CMP TAB_VER,Y      correspond au
                        secteur 0 ?
598      BEQ VER_L2
599      SEC              ; erreur: il faut
                        réduire GAP3
600      RTS
601 VER_L3 CLC
602      RTS
603 * (on peut maintenant demander à ProDOS de
      contrôler
604 * l'ensemble de la piste, sans risque
      qu'il se plante...)
605
606 VER_4_4 PHA
607      LSR
608      ORA $5AA
609      STA TAB_VER,Y      initialise table
                        de vérification
610      PLA
611      ORA $5AA
612      INY
613      STA TAB_VER,Y
614      RTS
615
616
617 *****
618 * 4 - FORMATEUR LOGIQUE *
619 *****
620
621 FORMAT_L = *
622 * (en entrée tous les blocs ont été
      initialisés à zéro binaire)

```


D5, AA, 96), des informations sur les numéros de volume, de piste, de secteur et un contrôle de somme (chacune codées sur deux octets), et un suffixe (DE, AA, EB) ;

- la zone données comprend un préfixe (D5, AA, AD), les 342 octets représentant les 256 octets du secteur, un contrôle de somme sur un octet, et un suffixe (DE, AA, EB) ;
- chaque zone est précédée d'un certain nombre "d'octets" spéciaux à dix bits, les huit premiers à un, les deux suivants à 0 ; ces octets servent à synchroniser la lecture sur le premier bit d'un octet, et non pas sur une position quelconque dans un octet ;
- pour adapter le formatage à la vitesse de rotation, il suffit de "jouer" sur le nombre d'octets de synchronisation entre deux secteurs ;

Ces contraintes doivent être gérées par le logiciel d'accès disque, et donc par le programme de formatage.

La difficulté majeure à programmer cette routine réside dans sa dépendance au temps : chaque bit doit être écrit en quatre

cycles-machine (soit trente-deux cycles pour un octet normal, et quarante cycles pour un octet de synchronisation), et il est donc impératif de compter le nombre de cycles de chaque instruction, car tout décalage rend la disquette illisible. Le développement a été grandement facilité par l'utilisation de Bugbyter (Pom's 13).

En conséquence et afin de rendre la lecture du source plus aisée, les instructions assembleur qui ne servent apparemment à rien mais permettent, en fait, de "passer le temps" ont été regroupées dans des macro-instructions de temporisation.

ProDOS, comme le DOS 3.3, Pascal ou CP/M, utilise le même formatage physique en 16 secteurs. En revanche le formatage logique diffère pour chaque système.

Formatage logique

Le formatage logique est l'organisation interne du système d'exploitation qui rend transparent, à l'utilisateur final, l'éparpillement en blocs d'un fichier sur la disquette ; (le bloc est une unité logique regroupant

deux secteurs).

Dans le cas de ProDOS, le formatage logique implique les actions suivantes :

- mise à zéro binaire de tous les blocs.
- les blocs deux à cinq contiennent le catalogue du volume ; ils sont chaînés entre eux par des pointeurs avant et arrière. De plus, le bloc 2 contient des informations générales sur la disquette, tel que le nom du volume et sa date de création.
- le bloc 6 contient la carte du volume ; après une initialisation, les sept premiers blocs doivent être marqués occupés.
- les blocs 1 et 2 contiennent les instructions machine qui chargent ProDOS. Si la disquette est destinée à être bootée, il faudra copier le fichier ProDOS sur cette disquette, ainsi qu'un interpréteur système (généralement Basic.System). Dans le cas contraire, la disquette ne pourra être bootée (cas d'une disquette de données), et ProDOS affichera un message d'erreur en cas de tentative.

Ces deux premiers blocs diffèrent selon les versions de ProDOS ;

```

623 LDA #981
624 STA OP_MLI      on appelle
                   ProDOS en
                   écriture

625 LDY #0
626 STY M_BLOCK+1
627
628 * 4.1 - Blocs 3 à 5 du Volume Directory
629 * -----
630 LDA #5          dernier bloc du
                   Volume Directory
631 STA M_BLOCK
632 LDA #4          le précédent est
                   le bloc 4
633 STA (HIMEM).Y  Himem pointe vers
                   le buffer général
634 JSR WRITE_BL
635
636 DEC M_BLOCK     on passe au bloc 4
637 LDA #3
638 STA (HIMEM).Y
639 LDA #5
640 LDY #2
641 STA (HIMEM).Y
642 JSR WRITE_BL
643
644 DEC M_BLOCK     ...puis on bloc 3
645 LDA #4
646 STA (HIMEM).Y
647 LDY #0
648 LDA #2
649 STA (HIMEM).Y
650 JSR WRITE_BL
651
652 * 4.2 - Bloc du Volume Bit Map
653 * -----
654 LDA #5FF        bit à 1 indique

```

```

655 LDY #522        bloc libre
                   ($22+1) * 8 - 280
                   blocs sur disq

656 WRITE_L2 STA (HIMEM).Y
657 DEY
658 BNE WRITE_L2
659 LDA #1           les blocs 0 à 6
                   sont utilisés
660 STA (HIMEM).Y
661 LDA #6          la Bit Map est
                   sur le bloc 6
662 STA M_BLOCK
663 JSR WRITE_BL
664
665 * 4.3 - Premier bloc du Volume Directory
666 * -----
667 TIA             ;=0
668 WRITE_L3 STA (HIMEM).Y on remet le bloc
                   à zéro binaire
669 INI
670 BNE WRITE_L3
671
672 LDA #3
673 LDY #2
674 STA (HIMEM).Y  pointeur avant
                   prochain bloc
675
676 INY
677 INY
678 LDA #5F0        Directory du Volu
679 ORA NOM_LG      lg nom du volume
                   (0-5F)
680 STA (HIMEM).Y
681
682 LDX #0
683 WRITE_L4 LDA NOM_VOL,X nom du volume
684 INY

```

```

685 STA (HIMEM).Y
686 INX
687 DEC NOM_LG
688 BNE WRITE_L4
689
690 JSR DATETIME     appel à une
                   éventuelle
                   horloge
691
692 LDX #3
693 LDY #51F
694 WRITE_L5 LDA DATE,X
695 STA (HIMEM).Y   met date/heure
                   création disq
696
697 DEY
698 DEX
699 BPL WRITE_L5
700
701 LDX #8
702 LDY #52A
703 WRITE_L6 LDA VOL_DIR,X met constantes
                   initiales du
                   Directory
704
705 STA (HIMEM).Y
706 DEY
707 DEX
708 BPL WRITE_L6
709
710 LDA #2           1er bloc du
                   Directory est #2
711 STA M_BLOCK
712 JSR WRITE_BL
713
714 * 4.4 - Blocs génériques du boot
715 * -----
716 LDX VC_BOOT2+1
717 STX M_BUFFER
718 LDX VC_BOOT2+2

```


aussi, plutôt que rendre le programme de formatage dépendant d'une version précise (et doubler la taille du source), utilisez INIT.LINK pour configurer la routine à votre propre version de ProDOS.

Afin de contrôler le bon déroulement de l'initialisation physique de la disquette, un double contrôle est effectué après le formatage de chaque piste. Après avoir initialisé le dernier secteur d'une piste, on vérifie que le prochain secteur lu est bien le secteur zéro ; ceci permet de s'assurer que l'on n'a pas écrasé le début de la piste en écrivant la fin (cas d'un lecteur trop rapide), et que l'on n'a pas laissé un ancien secteur entre le début et la fin (cas d'un lecteur trop lent). Deuxième contrôle, plus approfondi celui-ci, on demande à ProDOS, par des appels à la fonction READ_BLOCK, de vérifier que tous les secteurs de la piste peuvent être correctement lus. Attention : si on demandait à ProDOS de lire une disquette mal formatée physiquement, celui-ci se "planterait" lamentablement en faisant "trépigner" le bras de lecture... un bug ?

Cette description vous a montré la complexité des méthodes

d'accès-disque lorsqu'elles sont analysées au niveau le plus fin (et, en corollaire, la grande diversité des méthodes de protection possibles...).

Avec un peu de pratique, il est possible d'accroître la capacité des disquettes 5 pouces, peut-être d'environ 10%, en n'utilisant qu'un seul secteur par piste, secteur de longueur variable selon que la piste se trouve près de l'axe de rotation ou des bords, mais il faudrait alors réécrire quasi-complètement le système d'exploitation et on se retrouve très vite coincé par le hardware !

Cette première approche vous a néanmoins, je l'espère, montré que l'utilisation de ProDOS en assembleur était beaucoup plus aisée que sous DOS 3.3, pourvu que le maître-mot "coexistence" soit respecté entre les différents programmes.

Bibliographie

ProDOS Technical Reference Manual (Apple)
Beneath Apple ProDOS (Quality Software)



Programme CMDLOAD.S (Assembleur Big Mac)

```

1      LST OFF
2 *
3
4 *****
5 *
6 *          CMDLOAD
7 *
8 *****
9
10 *****
11 * Copyright (C) 1985 Alexandre Avrane *
12 *
13 * Modifié: 06/04/85
14 * Créé: 10/03/85
15 *
16 * Assembleur: Big Mac
17 *****
18
19
20 * Objectifs:
21 * -----
22 * Installe une commande externe
  ProDOS/Basic System
23 * Environnement: ProDOS : Basic System
24 * Compatibilité: ProDOS B.4, 1.0, 1.0.1 et
  suivantes
25 * Intégration: chargé en $2000
26 * Restrictions: aucun fichier ouvert,
27 * chaînes de caractères Basic détruites
28
29 * Source (Big Mac): CMDLOAD.S
30 * Objet:          CMDLOAD
31
32 * Ce programme se charge en $2000
33 * Avant son exécution, la commande externe
  doit avoir
34 * été chargée et répondre aux conventions
  suivantes:
35 * -chargement en $2100
36 * -octet haut de l'adresse de fin
  incrémenté de 1 en $2102
37 * -octet haut de la longueur en $2104
38 * -adresse de la commande externe
  précédente en $2106.2107
39
40 * Adresses utilisées:
41 * -----
42 LENGTH = $2F          longueur-1 d'une
                          instruction 6502
43 A1      = $3C          vecteurs du MOVE
44 A2      = $3E
45 A4      = $42
46 PTR     = $48          vect temporeire
47 HIMEM   = $73
48
49 OFFSET  = A1
50 FLAG MAJ = A1+1
51 BASE    = $2000        ad de chargement
52 START   = BASE+$100
53 HFIN    = BASE+$102
54 HLONG   = BASE+$104
55 OLDCMD  = BASE+$106
56
57 EXTCHD  = $BE06        vecteur entrée
                          vers adr externe
58 ERRORT  = $BE09        vecteur erreur
59 OPENCNT = $BE4D        nb fichier ouvert
60 MLI      = $BF00        entrée de ProDOS
61 BITMAP  = $BF58        carte des pages
                          RAM utilisées
62
63 OPCODE  = $F88E        fournit long
                          instruction 6502
64 COOT    = $FD8D        routine affichage
65 MOVE    = $FE2C        déplacent mémoire
66
67 ORG     BASE
68
69 *****
70 * 1 - INITIALISEUR
71 *****
72
73      LDA MLI
74      CMP #$4C           'jmp' ?
75      BEQ VER_RAM
76      LDA #$87           ProDOS pas actif:
                          on arrête
                          en "beepant"
77      JMP COOT
78
79 * 1.1 - Recherche de la mémoire

```

```

716      STX M_BUFFER+1
717      DEC M_BLOCK      d'abord le bloc 1
718      JSR WRITE_ML
719      DEX
720      DEX
721      STX M_BUFFER+1
722      DEC M_BLOCK      puis le bloc 0
723      JMP WRITE_ML
724
725 * 4.a - Module d'écriture d'un bloc
726 * -----
727 /2/ WRITE_ML JSR GO_MLI
728      BCC WRITE_BZ
729      PLA              ;si erreur on
                          saute un niveau
                          d'appel
730
731      PLA
732      WRITE_BZ RTS
733 *****
734 * 5 - ZONES DE TRAVAIL
735 *****
736
737 VC_PARM LDA M_PARM      instructions pour
                          vecteurs relogeab
                          ;termine la zone
                          de codes 6502 rel
738      BRK
739
740 * 5.1 - Constantes & variables
741 * -----
742 COMMAND ASC 'INIT'
743 FTRACK DFB 0
744 FSECTOR DFB 0
745 FSLOT DFB 0
746 A_TRACK DFB 0

```

```

747 D_TRACK DFB 0
748 ARM_FLAG DFB 0
749 ARM_DIFF DFB 0
750 ARM_TAB DFB 2,4,6,0,6,4,2,0
751 GAP3_BIS DFB 0
752 RETRYCNT DFB 0
753 NOM_LG DFB 0
754 NOM_VOL DFB B,B,B,B,B,B,B,B,
              B,B,B,B
              (15 blancs)
755 TAB_VER DFB $D5,$AA,$96,$AA,$AB,0,0,
              $AA,$AA,0,0,$DE,$AA
756 VOL_DIR DFB $C3,$27,$D,0,0,6,0,$18,1
757 SAVE_ERR DA 0
758
759 * 5.2 - Appel à ProDOS
760 * -----
761 GO_MLI JSR MLI
762 OP_MLI DFB $80          lecture/écriture
763 VC_MLI DA M_PARM
764      RTS
765 M_PARM DFR 3
766 M_INIT DFB 0
767 M_BUFFER DA 0
768 M_BLOCK DA 0
769
770 * 5.3 - Blocs du boot
771 * -----
772 * Zone mémoire chaînée par INIT.LINK
773 BOOT_BLK = **$200
774
775 FIN      = **400
776 LONG    = **START+$400
777      LST ON
778      END

```



```

80 * -----
81 VER_RAM LDA OPENCNT fichiers ouverts?
82 BEQ FIND_RAM
83 LDA #915 FILE STILL OPEN
84 JMP ERRROUT oui: impossible
                    placer la routine

85 FIND_RAM LDA HLONG
86 ADC #0 (carry=1)
87 JSR GETBUFR cherche pages RAM
                    consécutives
88 BCC RAM_OK ok, c'est trouvé
89 NO_RAM LDA #14 PROGRAM TOO LARGE
90 JMP ERRROUT
91
92 RAM_OK CMP HFIN Acc = msb 1ère
                    page allouée
93 BCC NO_RAM
94
95 * 1.2 - Installe la nouvelle commande
96 * -----
97 LDX EXTEND+2
98 STA EXTEND+2
99 STX OLDCMD+1
100 LDX EXTEND+1 et sauvegarde
                    l'ancien vecteur
101 STX OLDCMD de commande est
102 INY #0
103 STY EXTEND+1
104
105 * 1.3 - Réassemble le code objet
106 * -----
107 PHA
108 SEC #>START (c-1)
109 STA OFFSET
110 PLA
111 SEC
112 SEC #4 4 pages = 1 K
113 STA HIMEM+1 fin Himem et le
                    buffer général
114 LDA #>START
115 STA PTR+1
116 STY PTR START doit débu
                    ter sur une page
117
118 RELOC_L1 LDY #0
119 LDA (PTR),Y
120 BEQ MOVE_L1 BRK(000) sépare
                    code et constante
                    demande la long
121 JSR OPCODE d'une instruction
122 LDY LENGTH si 2 alors 3 oct
123 CPY #2
124 BNE RELOC_L2
125 LDA (PTR),Y prend octet haut
126 CMP #>START
127 BCC RELOC_L1 adresse plus
                    basse que nous
128 CMP HFIN
129 BCS RELOC_L2 adresse plus
                    haute que nous
130 ADC OFFSET ajoute décalage
131 STA (PTR),Y met à jour instru
132 RELOC_L2 LDA PTR
133 SEC
134 ADC LENGTH
135 STA PTR
136 LDA PTR+1
137 ADC #0
138 STA PTR+1
139 BNE RELOC_L1 =jmp
140
141 * 1.4 - Déplace code vers zone disponible
142 * -----
143 MOVE_L1 LDY #<START doit être 0 (code
                    aligné)
144 LDA #>START
145 STY A1
146 STA A1+1
147 CLC
148 ADC HLONG
149 STY A1
150 DEY
151 STY A2
152 STA A2+1
153 LDA EXTEND+2
154 STA A4+1
155 INY
156 JMP MOVE déplace, termine
                    l'initialisation

157
158
159 * 1.5 - recherche de pages libres en RAM
160 * -----
161 GETBUFR = *
162 * (ce module devait être intégré dans
ProDOS à l'adresse $BEE5
163 * en entrée: A = nombre de pages désirées
164 * X = 0 : alloc mémoire temporaire

```

```

165 * X > 0 : allocation permanente sur
                    Memory Bit Map
166 * en sortie: carry = 1 si erreur,
167 * sinon A = numéro 1ère page allouée
168
169 STA PAGE_N nb de pages
                    cherchées
170 LDA HIMEM+1
171 CLC
172 ADC #4 cherche a/c du
                    buffer de 1K
173 STA PAGE_1
174 STX FLAG_MAJ non nul: 2 passag
175 GBUF_L1 DEC PAGE_1 tête de série
176 SEQ GBUF_L6 pas trouvé
177 GBUF_L2 LDA PAGE_1
178 STA PAGE_2
179 GBUF_L3 LDA PAGE_2 page courante
180 PHA
181 LSR
182 LSR
183 LSR

```

```

191 LDA FLAG_MAJ
192 BNE GBUF_L4 non nul, 1er
                    passage sans MAJ
193 LDA BITMASK,Y
194 ORA BITMAP,X MAJ carte mémoire
195 STA BITMAP,X
196 GBUF_L4 LDA PAGE_1
197 SEC
198 DEC PAGE_2
199 SBC PAGE_2
200 CMP PAGE_N toutes pages
                    vérifiées?
201 BNE GBUF_L3 non, on contrôle
                    la page suivante
202 LDA FLAG_MAJ
203 BNE GBUF_L5 si non nul, 2ème
                    passage
204 CLC
205 LDX PAGE_2 c'est fini
206 INX
207 TAX
208 RTS

```

Pom's vous propose

"Dominos"

Thierry Haurie

Apple II+, IIe, IIC

Il est inutile de présenter le jeu de dominos; celui-ci bénéficie d'un graphisme très soigné (en couleur si vous disposez d'une carte "Chat Mauve") et les messages transmis par le programme sont, au choix, en Français, en Italien, en Allemand ou en Anglais.



80.00 F TTC franco
Bon de commande page 74

```

184 TAX ;multiplié par 8
185 DTA
186 AND #7 forme indice dans
                    masque de chq oct
187 TAY
188 LDA BITMASK,Y car Bit addr,X
189 AND BITMAP,X n'est possible
                    que sur IIC
190 BNE GBUF_L1 pas de place

209 GBUF_L5 LDA #0
210 STA FLAG_MAJ
211 BEQ GBUF_L2 =jmp
212 GBUF_L6 SEC
213 RTS
214 BITMASK DFR $80,$40,$20,$10,$8,$4,$2,$1
215 PAGE_N DFR 0
216 PAGE_1 DFR 0
217 PAGE_2 DFR 0
218 END

```


Récapitulation

CMDLOAD

2000- AD 00 BF C9 4C F0 05 A9
 2008- 87 4C ED FD AD 4D BE F0
 2010- 05 A9 15 4C 09 BE AD 04
 2018- 21 69 00 20 98 20 90 05
 2020- A9 0E 4C 09 BE CD 02 21
 2028- 90 F6 AE 08 BE 8D 08 BE
 2030- 8E 07 21 AE 07 BE 8E 06
 2038- 21 A0 00 8C 07 BE 48 E9
 2040- 21 85 3C 68 38 E9 04 85
 2048- 74 A9 21 85 49 84 48 A0
 2050- 00 B1 48 F0 27 20 8E F8
 2058- A4 2F C0 02 D0 0F B1 48
 2060- C9 21 90 09 CD 02 21 B0
 2068- 04 65 3C 91 48 A5 48 38
 2070- 65 2F 85 48 A5 49 69 00
 2078- 85 49 D0 D3 A0 00 A9 21
 2080- 84 3C 85 3D 18 6D 04 21
 2088- 84 42 88 84 3E 85 3F AD
 2090- 08 BE 85 43 C8 4C 2C FE
 2098- 8D FB 20 A5 74 18 69 04
 20A0- 8D FC 20 86 3D CE FC 20
 20A8- F0 47 AD FC 20 8D FD 20
 20B0- AD FD 20 48 4A 4A 4A AA
 20B8- 68 29 07 A8 B9 F3 20 3D
 20C0- 58 BF D0 E1 A5 3D D0 09
 20C8- B9 F3 20 1D 58 BF 9D 58
 20D0- BF AD FC 20 38 CE FD 20
 20D8- ED FD 20 CD FB 20 D0 D0
 20E0- A5 3D D0 07 18 AE FD 20
 20E8- E8 8A 60 A9 00 85 3D F0
 20F0- B9 38 60 80 40 20 10 08
 20F8- 04 02 01 00 00 00 15

2190- B1 48 AA CA E0 10 B0 2C
 2198- 8E 29 25 C8 B1 48 C9 2F
 21A0- D0 22 C8 B1 48 C9 41 90
 21A8- 1B C9 5B B0 17 99 28 25
 21B0- CA F0 15 C8 B1 48 C9 2E
 21B8- F0 F3 C9 30 90 E7 C9 3A
 21C0- 90 EB B0 E1 A9 40 D0 C0
 21C8- 20 D3 21 B0 BB 20 58 24
 21D0- B0 B6 60 AD 61 BE 0A 0A
 21D8- 0A 0A 8D 1A 25 0D 62 BE
 21E0- AA CA BD 8A C0 20 43 23
 21E8- BD 80 C0 BD 82 C0 BD 84
 21F0- C0 BD 86 C0 A9 19 8D 27
 21F8- 25 A9 80 8D 54 25 A9 10
 2200- 8D 28 25 EE 27 25 A9 00
 2208- 8D 19 25 78 20 47 22 B0
 2210- 32 20 26 24 58 90 0F CE
 2218- 27 25 AD 27 25 C9 08 B0
 2220- E5 A9 27 38 B0 1D 20 EA
 2228- 23 90 07 CE 28 25 D0 D6
 2230- F0 EF AD 18 25 EE 18 25
 2238- AC 18 25 20 53 23 C9 23
 2240- 90 BC 18 BC 88 C0 60 A0
 2248- 05 AD 18 25 48 20 49 24
 2250- 68 49 01 A0 09 20 49 24
 2258- BD 89 C0 38 A9 2B BC 8D
 2260- C0 BC 8E C0 30 E0 48 68
 2268- A9 FF 9D 8F C0 DD 8C C0
 2270- A0 FF EA EA EA EA 20 CC
 2278- 23 A9 D5 20 C2 23 A9 AA
 2280- 20 C2 23 A9 96 20 C2 23
 2288- A9 01 EA 20 D1 23 AD 18
 2290- 25 20 B1 23 AD 19 25 20
 2298- B1 23 AD 18 25 4D 19 25
 22A0- 49 01 48 4A 09 AA 9D 8D
 22A8- C0 DD 8C C0 68 09 AA 20
 22B0- C1 23 A9 DE 20 C2 23 A9
 22B8- AA 20 C2 23 A9 EB 20 C2
 22C0- 23 A0 05 20 D7 23 A9 D5
 22C8- 20 C2 23 A9 AA 20 C2 23
 22D0- A9 AD 20 C2 23 A0 56 D0
 22D8- 03 EA EA EA A9 96 EA EA
 22E0- EA EA EA 9D 8D C0 DD 8C
 22E8- C0 88 D0 ED F0 01 EA 20
 22F0- 58 FF EA EA 9D 8D C0 DD
 22F8- 8C C0 88 D0 F1 A9 96 EA
 2300- 20 C2 23 A9 DE 20 C2 23
 2308- A9 AA 20 C2 23 A9 EB 20
 2310- C2 23 A9 FF 20 C2 23 AC
 2318- 27 25 20 CE 23 EE 19 25
 2320- EA EA 48 68 9D 8D C0 DD
 2328- 8C C0 4C 2D 23 AD 19 25
 2330- C9 10 B0 03 4C 79 22 48
 2338- 68 48 68 BD 8E C0 BD 8C
 2340- C0 18 60 AE 1A 25 BD 89
 2348- C0 BD 8F C0 BD 8C C0 A9
 2350- 30 A0 00 8D 1B 25 8C 1C
 2358- 25 A9 00 8D 1D 25 AD 1B
 2360- 25 38 ED 1C 25 F0 30 B0
 2368- 04 49 FF 69 01 8D 1E 25
 2370- 2E 1D 25 4E 1B 25 2E 1D

2378- 25 0E 1D 25 AC 1D 25 B9
 2380- 1F 25 20 A1 23 B9 20 25
 2388- 20 A1 23 98 49 02 A8 CE
 2390- 1E 25 AD 1E 25 D0 E8 AD
 2398- 1C 25 8D 18 25 AE 1A 25
 23A0- 60 0D 1A 25 AA BD 81 C0
 23A8- A9 57 20 A8 FC BD 80 C0
 23B0- 60 48 4A 09 AA 9D 8D C0
 23B8- DD 8C C0 68 EA EA EA 09
 23C0- AA EA EA 48 68 9D 8D C0
 23C8- DD 8C C0 60 D0 06 D0 0B
 23D0- EA 20 58 FF 4C D7 23 EA
 23D8- 4C DB 23 EA A9 FF 9D 8D
 23E0- C0 DD 8C C0 88 D0 E9 EA
 23E8- EA 60 A9 00 8D 5D 25 8D
 23F0- 5A 25 A5 74 8D 5B 25 AD
 23F8- 12 25 8D 56 25 AC 62 BE
 2400- 88 98 4A 6A 0D 1A 25 8D
 2408- 59 25 A0 08 AD 18 25 0A
 2410- 0A 0A 8D 5C 25 90 03 EE
 2418- 5D 25 20 51 25 B0 06 EE
 2420- 5C 25 88 D0 F5 60 A0 01
 2428- BD 8C C0 C9 FF D0 F9 88
 2430- 10 F6 BD 8C C0 10 FB C9
 2438- FF F0 F7 C8 C0 0D F0 07
 2440- D9 39 25 F0 ED 38 60 18
 2448- 60 48 4A 09 AA 99 39 25
 2450- 68 09 AA C8 99 39 25 60
 2458- A9 81 8D 54 25 A0 00 8C
 2460- 5D 25 A9 05 8D 5C 25 A9
 2468- 04 91 73 20 08 25 CE 5C
 2470- 25 A9 03 91 73 A9 05 A0
 2478- 02 91 73 20 08 25 CE 5C
 2480- 25 A9 04 91 73 A0 00 A9
 2488- 02 91 73 20 08 25 A9 FF
 2490- A0 22 91 73 88 D0 FB A9
 2498- 01 91 73 A9 06 8D 5C 25
 24A0- 20 08 25 98 91 73 C8 D0
 24A8- FB A9 03 A0 02 91 73 C8
 24B0- C8 A9 F0 0D 29 25 91 73
 24B8- A2 00 BD 2A 25 C8 91 73
 24C0- E8 CE 29 25 D0 F4 20 06
 24C8- BF A2 03 A0 1F BD 90 BF
 24D0- 91 73 88 CA 10 F7 A2 08
 24D8- A0 2A BD 46 25 91 73 88
 24E0- CA 10 F7 A9 02 8D 5C 25
 24E8- 20 08 25 AE 09 21 8E 5A
 24F0- 25 AE 0A 21 8E 5B 25 CE
 24F8- 5C 25 20 08 25 CA 8E
 2500- 5B 25 CE 5C 25 4C 08 25
 2508- 20 51 25 90 02 68 68 60
 2510- AD 58 25 00 49 4E 49 54
 2518- 00 00 00 00 00 00 02
 2520- 04 06 00 06 04 02 00 00
 2528- 00 00 20 20 20 20 20
 2530- 20 20 20 20 20 20 20
 2538- 20 D5 AA 96 AA AB 00 00
 2540- AA AA 00 00 DE AA C3 27
 2548- 0D 00 00 06 00 18 01 00
 2550- 00 20 00 BF 80 58 25 60
 2558- 03 00 00 00 00 00 20

Récapitulation

INIT.CODE0

2100- D8 A9 27 A9 08 AD 58 FF
 2108- AD 5E 27 AD 6C BE 85 48
 2110- AD 6D BE 85 49 A0 01 B1
 2118- 48 D9 13 25 D0 3F C8 C0
 2120- 05 90 F4 88 88 8C 52 BE
 2128- A9 00 8D 0F BF 8D 53 BE
 2130- A9 01 8D 54 BE A9 04 8D
 2138- 55 BE AD 0D BE 8D 4F 25
 2140- AD 0E BE 8D 50 25 AD 61
 2148- 21 AD 47 21 8D 50 BE 8D
 2150- 0D BE AD 48 21 8D 51 BE
 2158- 8D 0E DE 18 60 30 6C 06
 2160- 21 AE 4F 25 8E 0D RE AE
 2168- 50 25 8E 0E BE C9 00 F0
 2170- 08 C9 06 F0 04 C9 08 D0
 2178- 12 AD 61 BE 09 C0 8D 85
 2180- 21 A9 28 AE FF C0 F0 06
 2188- 20 8B BE 4C 0C BE A0 00

Programme **TDUMP.CODE.S** **(Assembleur Big Mac)**

```

1      LST OFF
2 *
3
4 *****
5 *
6 *      PMOUVS TDUMP
7 *
8 *****
9
10 *****
11 * Copyright (C) 1985 Alexandre Avrane *
12 *
13 * Modifié: 26/04/85
14 * Créé: 07/04/85
15 *
16 * Assembleur: Big Mac
17 *****
18
19
20 * Objectifs:
21 *
22 * Fournit un dump hexadécimal/ASCII d'un
  fichier ProDOS
23 *
24 * Syntaxe:
25 *
26 * TDUMP <nom fichier> [,S <slot>] [,D
  <drive>]
27 *
28 * Environnement: ProDOS + Basic System
29 * Intégration: cde externe Basic System
30 *
31 * Moins de $100 octets
32 * Arrêt momentanée par une touche,
  définitif par ctrl-Q.
33 *
34 * Ce programme doit être chaîné avec
  CMDLOAD pour être
35 * initialisé comme une commande externe.
36 * Source (Big Mac): INIT.TDUMP.S
37 * Objet:      INIT.TDUMP
38
39 * Adresses utilisées:
40 *
41 PTR      = 548      vecteur tempora
42 HIMEM    = $73
43 READ_BUF = $200      buffer de lec-
  ture de 8/16 o.
44
45 PRINTERR = $BE0C      affiche erreurs
46 VECTOUT  = $BE30      routine sortie
47 XTADDR   = $BE50      vecteur sortie
  d'une cde exter
48 XTEN     = $BE52      lg d'une cde ex
49 XCNUM    = $BE53      n' cde ProDOS
50 PBITS    = $BE54      bits d'autori-
  sation paramétr
51 VPATH1   = $BE6C      vecteur vers
  nom volume
52 GOSYSTEM = $BE70      appel à ProDOS
  via Basic.Syste
53 SOPEN    = $BECB      table des para-
  mètres de OPEN
54 SREAD    = $BED5      table des para-
  mètres de READ
55 SCLOSE   = $BEDD      table des para-
  mètres de CLOSE
56 SYSERR   = $BF0F      code erreur MLI
57
58 KEYBOARD = $C000      clavier
59 STROBE   = $C010      ras clavier
60 PRBL2    = $F94A      aff x blancs
61 RDKEY    = $FD18      attente clavier
62 CROUT    = $FD8E      envoi d'un CR
63 PRNTXY3  = $FD99      affiche X et Y
  en hexa et "-"
64 PRBYTE   = $FDDA      affiche accumulé
65 COUT     = $FDED      sortie caractère
66 RTS      = $FF58      contient un RTS
67
68      ORG $2100      (CMDLOAD de
  $2000 à $20FF)
69
70 * 1.1 - Vérifie que la commande externe
  est la nôtre
71 *
72 START    CLD          ;nécessaire
73
74 * Conventions de link avec CMDLOAD:

```

```

75      LDA #>FIN+$100
76      LDA #>LONG-$100
77 V_OLDCMD LDA RTS
78
79      LDA VPATH1      pointeur vers
  cde actuelle
80
81      STA PTR
82      LDA VPATH1+1
83      STA PTR+1
84      LDY #1
85      LDA (PTR),Y      obtient un caractère
86      CMP COMMAND-1,Y  c'est "TDUMP"?
87      BNE NO_CMD      non
88      INY
89      CPY #5+1
90      BCC COMPAR
91
92 * 1.2 - Demande à PROBUS d'examiner la
  suite de la commande
93
94      DEY
95      DEY
96      STY XLEN      stocke lg - 1
97      LDA #0
98      STA SYSERR      initialise
  code erreur MLI
99      STA XCNUM      indique cde ext
100     LDA #$00000001
101     STA PBITS      nom de fichier/
  volume autorisé
102     LDA #$00000100
103     STA PBITS+1    slot & drive
  autorisés
104 V_SUITE LDA SUITE      vecteur de
  retour après
  examen
105
106     LDA V_SUITE+1
107     STA XTADDR      indique à Pro-
  DOS où revenir
108     LDA V_SUITE+2
109     STA XTADDR+1
110     CLC
111
112     RTS
113
114     NO_CMD SEC      ;cde n'était
  pas pour nous
115     JMP (V_OLDCMD+1) alimenté par
  CMDLOAD
116
117 * 1.3 - Ouverture du fichier
118 SUITE LDA VPATH1      nom fichier re-
  copié en param
119
120     STA SOPEN+1
121     LDA VPATH1+1
122     STA SOPEN+2
123     LDA HIMEM
124     STA SOPEN+3
125     STA SREAD+5      (acc=0)
126     STA OFFSET
127     STA OFFSET+1
128     LDA HIMEM+1
129     STA SOPEN+4
130
131     LDA #SCB      code OPEN
132     JSR GOSYSTEM
133     BCS ERRUR
134     LDA SOPEN+5      réf recopié...
  en paramètre
  du READ.
135     STA SREAD+1
136
137     STA SCLOSE+1    et du CLOSE
138
139     LDA #<READ_BUF utilise buffer
  $200 pour lire
140
141     STA SREAD+2
142     LDA #>READ_BUF
143     STA SREAD+3
144
145 * 1.5 - Boucle de lecture
146
147     LDX #16
148     LDA VECTOUT+1
149     AND #SF0      conserve le
  1er demi-octet
150     CMP #SC0      sort vers slot?
  oui: 80 col
151     BEQ COL
152     LDX #8
153     STX SREAD+4    on lira 8 ou 16
  octets/ligne
154     LDA #SA0      caractère blanc
155     STA READ_BUF-1,X
156     DEX

```

```

154     BNE BLANK
155
156     LOOP LDA #SCA      READ
157     JSR GOSYSTEM
158     BCC OK
159     CMP #5      fin du fichier?
  oui
160     BEQ FINAL
161     JMP PRINTERR
162
163     OK LDA OFFSET
164     LDY OFFSET+1
165     JSR PRNTXY3      affiche adresse
  depuis le début
166
167     LDX #1
168     JSR PRBL2
169
170     LDY #0
171     LDY $SREAD+4    nb d'oct à lire
172     LDA $READ_BUF,Y
173     AND #$7F
174     CMP #>20
175     BCS NORM
176     LDA #' '      remplace caract-
  ère de contral
  bit fort à 1
  pour COUT
177
178     JSR COUT
179     DEX
180     INY
181     CPY $SREAD+6
182     BCC LOOP1
183
184     INX
185     INX
186     LDY #0
187     BEQ CONT      -jmp
188     LDY #1
189     JSR PRBL2      affiche 1 blanc
190     LDA $READ_BUF,Y
191     JSR PRBYTE      affiche un oct
192     INY
193     CPY $SREAD+6
194     BCC LOOP2      dernier caract-
  ère non
195
196     JSR CROUT      fin de la ligne
197     LDA $SREAD+4    MAJ autres début
198     CLC
199     ADC OFFSET
200     STA OFFSET
201     LDA OFFSET+1
202     ADC #0
203     STA OFFSET+1
204
205     LDA KEYBOARD
206     BPL LOOP
207     STA STROBE
208     LDA #SA0
209     JSR RDKEY
210     CMP #591      ctrl-Q ?
211     BNE LOOP
212     LDA #SCC      CLOSE
213     JMP GOSYSTEM
214
215 * Variables
216
217     BRK
218     COMMAND ASC 'TDUMP'
219     OFFSET DA 0
220
221     FIN = +
222     LONG = FIN-START+$100
223     LST ON
224     END

```

Récapitulation **TDUMP.CODE**

```

2100- D8 A9 23 A9 01 AD 58 FF
2108- AD 6C BE 85 48 AD 6D BE
2110- 85 49 A0 01 B1 48 D9 17
2118- 22 D0 2D C8 C0 06 90 F4
2120- 88 88 8C 52 BE A9 00 8D
2128- 0F BF 8D 53 BE A9 01 8D
2130- 54 BE A9 04 8D 55 BE AD
2138- 4C 21 AD 38 21 8D 50 BE

```



```

2140- AD 39 21 8D 51 BE 18 60
2148- 38 6C 06 21 AD 6C DE 8D
2150- CC BE AD 6D BE 8D CD BE
2158- A5 73 8D CE BE 8D DA BE
2160- 8D 1D 22 8D 1E 22 A5 74
2168- 8D CF BE A9 C8 20 70 BE
2170- B0 36 AD D0 DE 8D D6 BE
2178- 8D DE BE A9 00 8D D7 BE
2180- A9 02 8D D8 BE A2 10 AD
2188- 31 BE 29 F0 C9 C0 F0 02
2190- A2 08 8E D9 BE A9 A0 9D
2198- FF 01 CA D0 FA A9 CA 20
21A0- 70 BE 90 07 C9 05 F0 6A
21A8- 4C 0C BE AE 1D 22 AC 1E
21B0- 22 20 99 FD A2 01 20 4A
21B8- F9 A0 00 AE D9 BE B9 00
21C0- 02 29 7F C9 20 B0 02 A9
21C8- 2E 09 80 20 ED FD CA C8
21D0- CC DB BE 90 E9 E8 E8 A0
21D8- 00 F0 02 A2 01 20 4A F9
21E0- B9 00 02 20 DA FD C8 CC
21E8- DB BE 90 EF 20 8E FD AD
21F0- D9 BE 18 6D 1D 22 8D 1D
21F8- 22 AD 1E 22 69 00 8D 1E

```

```

2200- 22 AD 00 C0 10 97 8D 10
2208- C0 A9 A0 20 18 FD C9 91
2210- D0 8B A9 CC 4C 70 BE 00
2218- 54 44 55 4D 50 00 00 20

```

Programme CMDLINK

```

1 REM * LINK COMMANDE EXTERNE *
10 D$ = CHR$(4)
15 TEXT : HOME : PRINT "LINK COMMAND
  E EXTERNE"
20 PRINT : INPUT "COMMANDE EXTERNE
  A CHAÎNER: ";C$
25 IF NOT LEN(C$) THEN END
27 PRINT "CHARGE "C$".CODE ..."
30 PRINT D$"BLOAD"C$.LOAD,A$2100"
35 A = PEEK(8452): REM $2104
37 PRINT "CHARGE CMDLOAD..."
40 PRINT D$"BLOAD CMDLOAD,A$2000"
44 ONERR GOTO 46
45 PRINT D$"CREATE"C$,T$F0"
46 POKE 216,0
48 PRINT "SAUVE "C$
50 PRINT D$"BSAVE"C$,A$2000,L"256"
  (A+2),T$F0"

```

Programme INIT.LINK

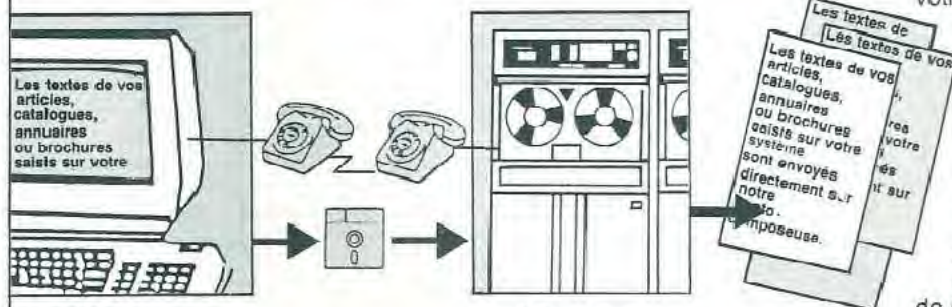
```

1 REM * LINK COMMANDE INIT *
10 D$ = CHR$(4)
15 TEXT : HOME : PRINT "LINK COMMAND
  E INIT"
19 PRINT : PRINT "CHARGE INIT.CODE0
  ..."
20 PRINT D$"BLOAD INIT.CODE0": REM
  $2100
23 FOR I = 0 TO 33: READ D: POKE 768 +
  I,D: NEXT
24 POKE 798, PEEK(8457): POKE 799,
  PEEK(8458) - 2
25 A = PEEK(8457) + 256 * PEEK(8458) -
  512: REM $2109
26 PRINT : PRINT "PLACEZ UNE DISQUE
  TTE PRODOS", "EN SLOT 6, DRIVE 1
  ->": GET Z$
30 CALL 770: IF PEEK(768) THEN PRINT
  "ERREUR...": END
40 B = A + 1024 - 2 * 4096
45 PRINT : PRINT "SAUVE INIT.CODE ..."
50 PRINT D$"BSAVE INIT.CODE,A$2100,
  L"A - 7424
90 DATA 0,1,32,0,191,128,28,3,141,0,3,
  208,14,238,32,3,238,31,3,238,31,3,20
  6,1,3, 16,231,96,3,96,0,0,0,0

```

Vos textes en direct de votre ordinateur à nos photocomposeuses

Gain de temps et économie



Les textes de vos articles, catalogues, annuaires ou brochures saisis sur votre micro-ordinateur sont envoyés directement sur notre photocomposeuse

Nous vous évitons ainsi, le coût et le temps de la saisie supplémentaire que nécessite le traitement traditionnel de la photocomposition avant l'impression des documents.

Si vous le désirez nous pouvons également nous charger de l'impression et du brochage.

Vous

Nous

TELECOMPO

(1) 328.18.63

PHOTOCOMPOSITION - BUREAUTIQUE - TRANSMISSION DE DONNÉES - GESTION DE FICHIERS - MATÉRIELS DE TRAITEMENT DE TEXTES
13 et 15, avenue du Petit-Parc - 94300 VINCENNES

Une référence : la revue
pom's

Les instructions de décalage et rotation du langage machine 6502 : ASL, LSR, ROL, et ROR, permettent bien des choses. Leurs fonctions sont les suivantes :

- ASL : décalage à gauche d'un octet. Tous les bits sont décalés d'une position vers la gauche. Le bit 7 va dans la retenue (C) et un zéro entre par le bit 0. Par exemple, si l'accumulateur contient la valeur %01010011 (\$53), une instruction ASL donnera 10100110 (\$A6), avec C=0.

- LSR : décalage à droite d'un octet. C'est le même principe, mais dans l'autre sens : les bits sont décalés vers la droite, C contient le bit 0 et le bit 7 est à zéro.

- ROL : Rotation à gauche d'un octet. L'effet est le même que ASL, à ceci près que le bit 0 prend la valeur de C avant le ROL : si l'accumulateur contient %01010011 (\$53) et la retenue est à 1, un ROL donnera %10100111 (\$A7). Il ne s'agit donc pas d'une vraie rotation, puisque le bit 7 sortant ne va pas remplacer le bit 0. Mais cela peut

être obtenu par :

```
CMP #$80 ; Bit 7 -> C
ROL
```

Le CMP #\$80 positionne C à 1 si l'accumulateur contient un octet supérieur ou égal à \$80, c'est-à-dire négatif, ou encore dont le bit 7 est à 1. Inversement, si cet octet est positif (bit 7 à zéro), C=0. Le ROL qui suit effectuera alors une vraie rotation. Cette méthode peut être utilisée pour coder des données sur une disquette, et empêcher d'éventuels curieux de les lire et les modifier avec un utilitaire genre DISKFIXER, CIA ou MOBBY DISK. L'utilisation d'un EOR #\$AA est devenue classique.

- ROR : Même principe, mais dans l'autre sens : l'octet est

décalé à droite, C entre par le bit 7, et le bit 0 sort et va dans C. Ici, effectuer une rotation complète est plus complexe, car on ne peut tester le bit 0 d'un octet directement. On peut faire :

```
PHA ;Sauvegarde
      l'accumulateur
LSR ;Bit 0 -> C
PLA ;Restaure l'acc. C
      contient toujours bit 0
ROR ;Et l'ex-bit 0 prend sa
      place au bit 7
```

Voir "Initiation à l'assembleur" (POM'S 14).

Les décalages ont de nombreuses applications. La principale, la multiplication entière, a déjà été traitée par Gérard Michel, ainsi que le test de bits.

Source ZOOM (Assembleur MerlIn)

```
1 *****
2 *
3 * AGRANDISSEMENT D'UNE *
4 * IMAGE HGR *
5 *
6 * (C) Pascal CANTOT *
7 * 04/05/85 *
8 *
9 *****
10
11
12 CNTR = $18
13 DLINE = $19
14 HBASE = $26
15 HMASK = $30
16 LINNUM = $50
17 XTAB = $FA
18 YTAB = $FB
19 HBASE1 = $FC
20 HBASE2 = $FE
21
22 CHKCOM = $DEBE
23 GETBYT = $E6F8
```

```
24 HPOSN - $F411
25
26
27 ORG $300
28 OBJ $300
29
30
31 GETPARM JSR CHKCOM
32 JSR GETBYT
33 CPX #21
34 BCS ILQERR
35 STX XTAB
36 JSR CHKCOM
37 JSR GETBYT
38 CPX #97
39 BCS ILQERR
40 STX YTAB
41
42 MAGNIFY LDA YTAB
43 CLC
44 ADC #95
45 STA CNTR
46 LDA #191
47 STA DLINE
48
49 MAGN1 LDA DLINE
50 JSR HPOSN
```

```
51 LDA HBASE
52 STA HBASE1
53 LDA HBASE+1
54 EOR #$60
55 STA HBASE1+1
56 DEC DLINE
57 LDA DLINE
58 JSR HPOSN
59 LDA HBASE
60 STA HBASE2
61 LDA HBASE+1
62 EOR #$60
63 STA HBASE2+1
64 DEC DLINE
65 LDA CNTR
66 JSR HPOSN
67 LDA HBASE
68 CLC
69 ADC XTAB
70 STA HBASE
71 LDY #19
72 MAGN2 LDA (HBASE),Y
73 PHA
74 AND #$10000000
75 STA HMASK
76 PLA
77 JSR DSHIFT
```


Les deux programmes présentés ici sont basés sur les décalages et les manipulations de bits : le premier les utilise pour compter des bits : il s'agit de FREDISK, qui donne le nombre de secteurs libres sur une disquette. Cela permet de voir s'il est possible de sauvegarder un fichier, mais aussi de contrôler facilement l'existence ou non d'un drive, ou la présence d'une mauvaise disquette. Le second illustre bien la puissance de ces instructions : il s'agit de ZOOM, agrandit une partie d'une image HGR au double de sa taille. On peut réaliser des effets de grossissement (voir DEMO ZOOM), mais aussi des génériques HGR courts et esthétiques.

FREDISK

Mode d'emploi

Après avoir fait "BRUN FREDISK", l'instruction USR est revectorisée. Il suffit alors d'un "PRINT USR(X)", où 'X' est le numéro du lecteur (1 ou 2). Tout autre valeur entraîne un "ILLEGAL QUANTITY ERROR") pour connaître le

nombre de secteurs libres sur la disquette se trouvant dans le lecteur X. En cas d'erreur de lecture (disquette vierge, par exemple), il y a un "Bip" et FREDISK retourne la valeur -1.

Le secteur 0 de la piste \$11 d'une disquette DOS 3.3 (VTOC de la disquette), contient, entre autres, une sorte de "carte" des secteurs (octets \$38 à \$C0), chaque secteur étant représenté par un bit et chaque piste par deux octets (les seize bits des seize secteurs qu'elle contient) plus deux octets à 0 (peut-être Apple avait-il prévu un éventuel doublement de la capacité de ses lecteurs).

Le principe de FREDISK est simple : grâce à la routine RWTS du DOS, il va lire ce secteur, puis compter les bits qui sont à 1 (=secteur libre) de l'octet \$38 à l'octet \$C0.

Voici la procédure : tout d'abord, on charge la VTOC en \$B3BB (emplacement qui lui est normalement réservé par le DOS) en tenant compte du lecteur spécifié. Ensuite, on prend une paire d'octets sur deux de \$38 à \$C0. On charge chaque octet dans l'accumulateur (routine GETBYTE) puis on appelle

SHIFT, qui retourne le nombre de bit à 1. Il suffit alors d'ajouter ce résultat à un compteur, puis de continuer jusqu'au 35ème octet. Alors, le compteur contiendra le nombre de secteurs libres.

SHIFT commence (ligne 85) par réinitialiser un compteur, BITCNTR, qui contiendra le nombre de bits qui sont à 1. Ensuite, une boucle est exécutée huit fois. A chaque fois, A est décalé. Le bit sortant va dans la retenue C. Puis, C est testé : si elle est à 1, on incrémente BITCNTR. Et on boucle huit fois...

A la fin, on a testé tous les bits de A et BITCNTR contient le nombre de bits qui étaient à 1.

Ce procédé est très utile, car il permet d'analyser les "BIT MAPS", comme une VTOC, une image HGR (compteur de collisions du Basic lors du dessin d'une forme), la MEMORY MAP du ProDOS, etc...

ZOOM HGR

Il s'agit d'un programme qui agrandit une image HGR, en mettant le résultat de cet agrandissement dans l'autre page

```

78      TYA
79      PHA
80      ASL
81      TAY
82      LDA  LINNUM
83      ORA  HMASK
84      STA  (HBASE1),Y
85      STA  (HBASE2),Y
86      TNY
87      LDA  LINNUM+1
88      ORA  HMASK
89      STA  (HBASE1),Y
90      STA  (HBASE2),Y
91      PLA
92      TAY
93      DEY
94      BPL  MAGN2
95
96      LDX  CNTR
97      BEQ  MAGNEND
98      DEX
99      STX  CNTR
100     CPX  YTAB
101     BCS  MAGN1
102
103     MAGNEND  RTS
104

```

```

105 *****
106
107 ILQERR  LDX  #53
108          JMP  $D412
109
110 DSHIFT  LDX  #0
111          STX  LINNUM
112          STX  LINNUM+1
113
114          LDX  #7
115 SHFT1   AS
116          BHP
117          JSR  ROLINNUM
118          PLP
119          JSR  ROLINNUM
120          DEX
121          BPL  SHFT1
122          JSR  ROLINNUM
123          LSR  LINNUM
124          RTS
125
126
127 *
128 ROLINNUM  ROL  LINNUM
129          ROL  LINNUM+1
130          RTS

```

Récapitulation ZOOM

```

0300- 20 BE DE 20 F8 E6 E0 15
0308- B0 7A 86 FA 20 BE DE 20
0310- F8 E6 E0 61 B0 6E 86 FB
0318- A5 FB 18 69 5F 85 18 A9
0320- BF 85 19 A5 19 20 11 F4
0328- A5 26 85 FC A5 27 49 60
0330- 85 FD C6 19 A5 19 20 11
0338- F4 A5 26 85 FE A5 27 49
0340- 60 85 FF C6 19 A5 18 20
0348- 11 F4 A5 26 18 65 FA 85
0350- 26 A0 13 B1 26 48 29 80
0358- 85 30 68 20 89 03 98 48
0360- 0A A8 A5 50 05 30 91 FC
0368- 91 FE C8 A5 51 05 30 91
0370- FC 91 FE 68 A8 88 10 DB
0378- A6 18 F0 07 CA 86 18 E4
0380- FB B0 A0 60 A2 35 4C 12
0388- D4 A2 00 86 50 86 51 A2
0390- 07 0A 08 20 A3 03 28 20
0398- A3 03 CA 10 F4 20 A3 03
03A0- 46 50 60 26 50 26 51 60

```


HGR (cela a plusieurs avantages : un zoom continu si on "switch" les pages APRES l'agrandissement, et la possibilité de réaliser quatre images HGR à partir d'une seule, en agrandissant successivement chaque quart de l'image.

Chaque point doublera en hauteur et en largeur, c'est à dire qu'il deviendra un carré de 2x2 pixels.

Pour lancer le programme, il suffit d'un "BLOAD ZOOM" (il se charge en \$300), puis d'un "CALL 768,X,Y" où X,Y sont les coordonnées du coin supérieur gauche du quart d'image à agrandir : X est le numéro d'octet, donc compris entre 0 et 20 (s'il est supérieur à 20, il y a une erreur, car le bord droit est alors en dehors de l'écran), et Y le numéro de la ligne (de 0 à 96).

Nous allons maintenant analyser le traitement que subit chaque octet de la page HGR doubler horizontalement (verticalement, il suffit de dédoubler la ligne).

Le problème est de dédoubler

chaque bit, pour obtenir le résultat sur deux octets, tout en conservant le bit 7 (couleur).

Ainsi, si on a un octet %10010100, il faudra retourner %10000110 10110000 ; analysons DSHIFT, à la ligne #110 du source :

A contient l'octet. Auparavant, HMASK a été mis à %10000000 ou %00000000 selon le bit de couleur. LINNUM (deux octets) est remis à 0. C'est lui qui contiendra le résultat. Puis SHFT1 est exécuté huit fois. A chaque fois, l'accumulateur est décalé, et le bit sortant va dans C. Le registre d'état (donc C également) est sauvegardé par PHP, et on appelle ROLINNUM, qui effectue le décalage sur 16 bits de LINNUM. La retenue entre donc dans LINNUM par le bit 0. Puis intervient la subtilité : on restaure la retenue (modifiée lors du décalage 16 bits) par PLP et on répète l'opération. Ainsi, le bit sortant de l'accumulateur se retrouve dédoublé dans LINNUM. Il reste à boucler huit

fois.

Ensuite, il s'agit de restaurer correctement le bit de couleur, car, bien que le zoom transforme tout point en couleur en point blanc par son dédoublement, il faut bien penser au fait que, pour l'Apple, il existe deux blancs et deux noirs différents, bien qu'ayant la même apparence. Aussi, il faut en tenir compte, au cas où l'utilisateur voudrait tracer des lignes sur l'image agrandie.

Le bit de couleur est dédoublé dans les bit 15 et 14 de LINNUM. Un appel à ROLINNUM le restreint au bit 15. Le bit de couleur du second octet est donc bien ajusté. Au deuxième maintenant... On décale vers la droite le premier octet (le bit 0, qui est "éjecté", vient de l'opération précédente), libérant ainsi le bit 7 (qui est alors à zéro). Ensuite, le programme principal effectue un ORA HMASK qui rétablit le bit de couleur original.



Programme DEMO ZOOM

```
20 ZOOM = 768
30 HOME : VTAB 24: HGR2
40 Charger ici une image graphique
   en $4000
50 FOR I = 1 TO 1000: NEXT : FOR
   I = 1 TO 3
60 FOR J = 1 TO 0 STEP -1
70 CALL ZO,5 + I * .5,40
80 POKE - 16300 + J,0: POKE
   230,32 + 32 * J
90 NEXT : NEXT : FOR I = 1 TO
   500: NEXT
100 POKE - 16301,0: INPUT "EN
   CORE (O/N) ? ";A$
110 IF ASC (A$) = 78 THEN TEXT
   : HOME : END
120 RUN
```

Récapitulation FREDISK

Pour utiliser FREDISK, faire
BRUN FREDISK

puis

A=USR(DRIVE)
avec DRIVE = 1 ou 2.

```
9500- A9 4C 85 0A A9 0D 85 0B
9508- A9 95 85 0C 60 20 FB E6
9510- 8A F0 09 E0 03 B0 05 8E
9518- EA B7 D0 03 4C 99 E1 A9
9520- BB 8D C3 AA A9 B3 8D C4
9528- AA 20 8D 95 90 0A A9 FF
9530- A0 FF 20 F2 E2 4C 3A FF
9538- A9 00 8D 8B 95 8D 8C 95
9540- A0 38 20 59 95 C8 20 59
9548- 95 C8 C8 C8 C0 C4 D0 F2
9550- AD 8C 95 AC 8B 95 4C F2
9558- E2 B9 BB B3 20 6F 95 18
9560- 6D 8B 95 8D 8B 95 AD 8C
9568- 95 69 00 8D 8C 95 60 8E
9570- 8A 95 A2 00 8E 89 95 A2
9578- 08 0A 90 03 EE 89 95 CA
9580- D0 F7 AD 89 95 AE 8A 95
9588- 60 00 00 00 00 A9 11 8D
9590- EC B7 A2 00 8E ED B7 8E
9598- EB B7 E8 8E F4 B7 A9 BB
95A0- 8D F0 B7 A9 B3 8D F1 B7
95A8- A9 B7 A0 E8 4C B5 B7
```

Source FREDISK (Assembleur Merlin)

```
1 *****
2 *
3 * F R E D I S K *
4 * ----- *
5 *
6 * Pascal CANTOT *
7 *
8 *****
9
10
11 USR = $0A
12 ADRVTOC = $AAC3
13 VTOC = $B3BB
14 GIVAYF = $E2F2
15 CONINT = $E6FB
16
17
18 ORG $9500
19 OBJ $9500
20
21 INITFRE LDA #$4C
22 STA USR
23 LDA #FREDISK
24 STA USR+1
25 LDA #>FREDISK
26 STA USR+2
27 RTS
```


Micro-informations

Jean-Michel Gourévitch

Ne drapeau de l'Apple // flotte au dessus de Cupertino. C'est la conséquence la plus visible de la réorganisation en forme de révolution de palais qui a secoué Apple. Exit Steve Jobs : avec la mise à l'écart du père fondateur, la priorité donnée à l'équipe chargée de développer le Macintosh a brusquement cessé. Les "parias", qui s'occupaient du // (qui fournit toujours à Apple l'essentiel de ses ressources) tiennent désormais le pouvoir. Sauront-ils modérer leur revanche, et ne pas étouffer les spécialistes du Mac ? De cette question dépendra probablement l'avenir d'Apple comme constructeur indépendant de micro-ordinateurs.

En attendant, c'est une véritable salve de nouveautés, concernant

l'Apple //, qui va probablement faire cet automne la joie des lecteurs de Pom's, toujours fanas de l'ancêtre, avec, pour commencer, le lecteur de disquettes (trois pouces et demi comme celles du Macintosh) qui viendra relayer l'antique lecteur de 143 Ko. Ce lecteur (de la taille du lecteur externe du Macintosh, et de la couleur de l'Apple //c) fabriqué par Sony (et baptisé Liron) est capable de stocker 800Ko sur les deux faces d'une disquette de trois pouces et demi. Les lecteurs de Pom's ont pu faire sa connaissance dans cette même rubrique du précédent numéro. L'appareil (utilisable sur Apple //c et //e) serait rebaptisé Unidisk 3.5. Prix espéré : moins de 4000

Francs. A noter qu'une version "Duodisk" (avec deux lecteurs et 1,6 Méga-octet) est utilisée par les développeurs travaillant sur produits Apple aux Etats-Unis.

La deuxième nouveauté attendue est une carte d'extension de mémoire. A l'image de la carte d'Applied engineering, mais vendue par Apple, elle permet d'étendre la mémoire vive de l'Apple //e jusqu'à 1 Méga-octet. Plus que le Mac !

Et ça n'est pas tout. La revue InfoWorld, généralement bien informée sur le dessous des cartes (d'extension) en a encore recensé d'autres. Celle notamment permettant de connecter l'Apple // au réseau Appletalk, lui ouvrant ainsi des possibilités de

Suite de Fredisk

28		56	STA	CNTR+1	88	LDX	#8
29 *****		57	LDY	#\$38	89		
30		58			90	SHFTLUP	ASL
31 FREDISK	JSR	59 SCANLUP	JSR	GETBYTE	91	BCC	SHIFT1
32	TXA	60	INY		92	INC	BITCNTR
33	BEQ	61	JSR	GETBYTE	93	SHIFT1	DEX
34	CPX	62	INY		94	BNE	SHFTLUP
35	BCS	63	INY		95		
36	STX	64	INY		96	LDA	BITCNTR
37	BNE	65	CPY	#\$C4	97	LDX	REGX
38 *		66	RNF	SCANLUP	98	RTS	
39		67			99		
40 FRERR	JMP	68	LDA	CNTR+1	100	BITCNTR	HEX 00
41		69	LDY	CNTR	101	REGX	HEX 00
42 RDVTOC	LDA	70	JMP	GIVAYF	102	CNTR	HEX 0000
43	STA	71			103		
44	LDA	72			104 *****		
45	STA	73 GETBYTE	LDA	VTOC, Y	105		
46	JSR	74	JSR	SHIFT	106 READVTOC	LDA	#\$11
47		75	CLC		107	STA	\$B7EC
48	BCC	76	ADC	CNTR	108	LDX	#0
49	LDA	77	STA	CNTR	109	STX	\$B7ED
50	LDY	78	LDA	CNTR+1	110	STX	\$B7EB
51	JSR	79	ADC	#0	111	INX	
52	JMP	80	STA	CNTR+1	112	STX	\$B7F4
53		81	RTS		113	LDA	#VTOC
54 SCANNER	LDA	82			114	STA	\$B7F0
55	STA	83 *			115	LDA	#>VTOC
		84			116	STA	\$B7F1
		85 SHIFT	STX	REGX	117	LDA	#\$B7
		86	LDX	#0	118	LDY	#\$E8
		87	STX	BITCNTR	119	JMP	\$D7D5

communication avec le Macintosh (c'est banal) et avec l'IBM PC (ça l'est moins).

Reste qu'avec leurs baisses de prix, les IBM et compatibles (et notamment le Tandy) sont pratiquement au même prix que l'Apple //. Comment concurrencer ces 16 bits ? Avec (probablement l'année prochaine), la version "16 bits" de l'Apple //, le vaisseau fantôme de la micro-informatique : tant de fois annoncé (notamment dans cette rubrique) et démenti avec la même constance par Apple. L'Apple 16 bits, un rêve ? Accrochez-vous : ce rêve est déjà

réalité. Grâce à deux cartes d'extension.

La première, la **MAX 816** de **Micro Magic**, contient un processeur 65816 cadencé à 4 Mhz. La carte contient en plus 256K de mémoire vive (extensible à 1 Méga-octet, mais le processeur peut adresser 16 Mégas). Pour faire fonctionner le processeur, **Micro Magic** a développé un système d'exploitation : le **MAX OS**, basé sur le système **UNIX** et prévoit d'ajouter à l'Apple une carte graphique (1024 x 1024 points) et un contrôleur permettant de gérer la mémoire vive en disque virtuel.

Une autre carte 65816 disponible, celle de **Com Log**, tourne moins rapidement (1 Mhz), mais pour 395 dollars seulement.

Autre nouveauté annoncée par **InfoWorld** : une imprimante **ImageWriter II** comprenant notamment un chargeur de papier automatique (fonctionnant aussi avec le Macintosh).

Pour lutter contre la concurrence, l'ensemble de la gamme Apple connaîtra des baisses de prix sensibles (aux Etats Unis, les prix aux revendeurs ont déjà baissé cet été de 4%. Ce ne serait qu'un début).

Et le Macintosh ?

S'intéressera-t-on encore au Macintosh chez Apple, avec cette avalanche de nouveautés pour l'Apple // ? Probablement.

Apple a fermé l'usine qui devait construire les disques durs. Il faudra donc que Cupertino ait recours à la concurrence pour fournir :

- un disque dur serveur (plus de 20 Mégas) pour l'utilisation au bureau avec le réseau **Appletalk**
- un disque dur de 10 à 20 Mégas pour le Macintosh (les pourparlers continuent avec **General Computer Co** pour l'éventuel montage de l'**Hyperdrive** dans le Mac, mais plusieurs fabricants sont en compétition).

Par contre, c'est chez Apple que sont actuellement mises au point de nouvelles ROM permettant la gestion d'informations stockées sur disques laser "compact disk" et une nouvelle version du **Finder**, encore plus rapide que la version 4.1. Les nouveaux lecteurs de disquettes double face (800Ko) seront eux aussi rapidement disponibles pour le Macintosh.

Par ailleurs, on étudie de nouvelles versions du Macintosh. Un Mac haut de gamme "ouvert" permettant l'insertion aisée de cartes d'extension et de disque dur serait sur la table à dessin,

ainsi que le "projet Jonathan" : un Mac avec écran couleur et modem intégré.

Une partie de toutes ces nouveautés serait distillée d'ici à la fin de l'automne. Le Mac haut de gamme, et l'Apple // 16 bits pourraient être officiellement annoncés en janvier, lors de la traditionnelle assemblée des actionnaires d'Apple. Ils seraient alors disponibles en 1986. "1986 sera l'année de l'introduction de nouveaux produits" a dit **Del Yocam**, l'ancien patron de la division Apple //, devenu responsable de toute la production d'Apple. Acceptons en l'augure.

Des logiciels pour l'Apple //

On se désespérerait, dit-on, chez Apple, devant le manque de logiciels créés pour l'Apple //. C'est vrai que les développeurs préfèrent apparemment travailler pour le Mac. Pourtant, tous n'ont pas abandonné l'ancêtre. **Version Soft** mettrait actuellement la dernière main à un incroyable "**Finder**", offrant sur l'Apple // toutes les facilités présentes habituellement à l'écran du Macintosh. Il ne resterait plus ensuite qu'à créer des logiciels exploitant ce **Finder**, et notamment un "**MacWrite**" pour l'Apple

// . **Version Soft** vient d'ailleurs de remporter un succès qui lui permet "d'assurer" : son logiciel de communications avec utilisation de la souris (**Version Com**) a été choisi par Apple pour être inclus dans une promotion comportant le modem **Sectrad** et un manuel de communications écrit par **Jacques Jean Bessières**.

Ordinateur Express, le distributeur de **Jane**, qui fut le premier programme intégré avec souris (hélas, trop léger), pour l'Apple // propose **Janet**, un petit programme pour apprendre la dactylographie, et étudie la possibilité d'améliorer **Jane** (formats de lettres pré-établies et de feuilles de calculs toutes faites, formats de gestion des cartes de crédit, date d'expiration des appareils ménagers, archives d'articles de presse etc... ainsi que l'élargissement de la feuille de calculs du tableur et des fiches du tableur).

Chez **Contrôle X**, on met aussi la dernière main à une gestion de fichiers facile à utiliser, employant notamment la souris.

De l'autre côté de l'Atlantique, les amateurs de programmation lorgnent avec intérêt sur le compilateur **C** de **Manx Software Systems**, (enfin disponible pour le **Prodos**). Cette firme met la dernière main à une version 65816 (pour l'Apple // 16 bits) de son compilateur **Aztec C**. Enfin, **Manx** propose une

gamme étendue de cross-compilateurs VAX, MS DOS, Macintosh, CP/M à destination de l'Apple II.

Broderbund se prépare à lancer **Fantavision**, un programme permettant de réaliser des dessins animés : on dessine des images (par exemple un bonhomme avec un bras tendu le long du corps et une autre le bras levé), et l'ordinateur se charge de réaliser toutes les images correspondant à la décomposition du mouvement. De tels programmes existaient déjà, mais pour des mini-ordinateurs valant plusieurs millions de dollars...

A remarquer, enfin, une nouvelle version du jeu d'échecs **Sargon III** d'**Hayden** utilisant la souris. L'Apple II ressemble, c'est déjà vrai, de plus en plus au Macintosh.

Des disques durs pour le Mac

En attendant celui "officiel" d'Apple, les disques durs se multiplient.

Le **Macbottom** de **Personal Computer Peripherals**, d'une capacité de 10 Méga-octet s'installe juste sous le Macintosh : sa taille correspond précisément à celle de la base du micro-ordinateur (son épaisseur est inférieure à 7 cm). Il se branche indifféremment sur la sortie modem ou imprimante du Mac. On peut le partitionner en volumes de taille variable, enfin, un "spooler" incorporé permet de continuer à travailler avec l'ordinateur pendant que l'imprimante fait son office.

Le **Paradise Mac 10** de **Paradise Systems** offre une capacité formatée de 10 Méga-octet, et se place à côté du Mac. Sa largeur : 3,5 pouces. Vitesse garantie : 5 fois plus vite qu'une disquette (vitesse de rotation : 3600 tours/minute). Prix : 1500 dollars.

Quant au **Sider** de **First Class Peripherals**, les lecteurs de Pom's avaient déjà lu sa description pour l'Apple II. Le

voici disponible pour le Macintosh : Technologie Winchester, 10 Méga-octet. Et surtout un prix "canon" : 695 dollars. L'appareil qui est vendu par correspondance aux Etats-Unis y fait un malheur. A ce prix, rien d'étonnant.

Remplacer la souris

On avait déjà tout essayé pour se débarrasser de la souris : joystick, crayon et tablette graphique, numériseur. On croyait avoir tout vu. Eh bien il manquait encore à l'appel le **VCS** (View Control System) de **Personics Corporation**. Quel bizarre ustensile : un casque qui s'installe sur le crâne de l'utilisateur du Macintosh. Un émetteur à ultra-sons s'installe au dessus du Macintosh. Il suffit ensuite de fixer du regard le point où l'on souhaite que le pointeur se déplace sur l'écran : le casque reçoit le signal ultra-sonore, les changements dans la rotation et l'angle de la tête de l'opérateur sont transmis au Macintosh. Pour cliquer, on utilise un bouton qui se colle sur le clavier, juste en dessous de la barre d'espace. Le produit paraît loufoque, mais ce n'est pas un poisson d'avril. Prix : 199 dollars.

Des logiciels pour le Mac

Avec, pour commencer, **Mac-Tell 2** de **Hello Informatique**. Cette nouvelle version permet toujours de transformer Macintosh en un Minitel intelligent, mais donne aussi la possibilité d'utiliser l'ordinateur comme terminal ASCII (et donc d'avoir accès à Calvados et au Club Apple) sans avoir à quitter le programme. Parmi les améliorations : un éditeur local permettant de lire, mettre en forme, ou rédiger un texte, à l'intérieur du programme, l'amélioration des procédures automatiques, la possibilité de transférer directement des

données dans Multiplan, l'envoi et la réception de fichiers (textes ou binaires), l'impression des procédures et textes édités, le réglage du mode de communication, une commande de suppression d'écho (pour la frappe des mots de passe), un mode veille (pour éviter le raccrochage automatique de Transpac), et une horloge affichée à l'écran avec chronomètre (au prix des communications, les mordus du Minitel apprécieront). En utilisation ASCII, on peut communiquer à 1200 bauds (même avec le modem Sectrad/Apple), on peut avoir accès aux procédures automatiques et au service de tracé de courbes Graphis de Calvados, configurer le clavier, régler l'affichage, sauvegarder l'écran ou le texte directement sur disque, imprimer en continu les lignes reçues, l'écran ou le texte sélectionné.

Bref, ce programme qui était déjà le meilleur outil émulant le Minitel sur Macintosh est désormais l'un des plus complets des programmes de communication.

Des logiciels graphiques

Et d'abord **VideoWorks** de **Hayden**. Un logiciel fantastique qui vous permettra de réaliser en deux coups de cuiller un dessin animé de plusieurs minutes. Ce dessin peut ensuite être sonorisé avec de la musique.

Les applications sont nombreuses : animation de vitrine, présentation animée d'une courbe, de l'évolution d'un produit, d'une tendance. En filmant avec une caméra l'écran du Macintosh, on peut réaliser un film animé.

Ce programme est vraiment capable de mettre en mouvement n'importe quoi. Il inclut aussi deux accessoires bien utiles : l'un, **Art Grabber**, installé dans le menu pomme, permet d'ouvrir un dessin MacPaint de l'intérieur du programme. Le second **Cheap Paint** permet de retoucher le dessin sans avoir à utiliser MacPaint. Prix : 79 dollars.

Vraiment pas cher par rapport aux possibilités de ce programme.

On peut aussi ouvrir des dessins de MacPaint de l'intérieur d'un programme avec **QuickPaint**, un petit utilitaire produit par **Enter Set**. Une fonction "Miniview" permet de consulter tous les dessins d'une disquette. On peut ensuite agrandir le dessin, puis la partie de dessin intéressante, et la couper ou la copier. Prix : 50 dollars. Le même éditeur vend aussi **QuickWord**, un utilitaire permettant de créer un glossaire dans MacWrite. Prix : 60 dollars.

Un autre programme permettant de réaliser des animations est produit par **Ann Arbor Software**, c'est **In Motion**. Les actions à réaliser sont représentées par des icônes. Le programme est livré avec **TotalPaint**, un outil de dessin comportant la plupart des fonctions de MacPaint, et bien sûr des sons, à ajouter aux graphiques.

Enfin, **PaintCutter** de **Silicon Beach Software** est l'un des outils inclus sur une disquette appelée **Accessory Pak 1**. **PaintCutter** permet d'ouvrir un dessin MacPaint sur toute la surface de l'écran (ce qui n'était hélas pas possible avec MacPaint). Une option permet de réaliser des écrans de démarrage, on peut aussi recopier dans un document MacWrite des documents plus larges que la surface de l'écran. Ce programme est utilisable avec le **Switcher**. Prix : 40 dollars.

Pour le travail sérieux

A noter : la nouvelle version (1.1) de **Think Tank 512K**. Cette version comporte des caractères différents, fonctionne avec le **Switcher**, et est désormais compatible avec **Word** et **MacWrite**.

Remarquable la nouvelle version d'**Overview de ProVue**. Cette base de données est désormais relationnelle, permet de classer dans l'ordre alphabétique un état de 20 pages en deux secondes, de

réaliser des graphiques. Les données peuvent être importées ou exportées de **MacWrite**, **Multiplan**, **MacTerminal**, **Lotus 123**, **Appleworks**, **dBase II**, etc..., une fonction **Clairvoyance** permet à l'ordinateur d'entrer correctement un mot ou un nom qu'on a déjà utilisé, en tapant seulement quelques unes de ses lettres : une sorte de glossaire automatique. Ce programme comporte aussi des macros commandant de réaliser des programmes automatiques. Bref, cette base de données relationnelles et automatisable est de la race des nouveaux programmes sortis en ce début d'automne, comme **Omnis 3**, **Quatrième dimension** et **Mac Base 500**. Son prix : 295 dollars.

Une rumeur persistante affirme que **Aston Tate**, le créateur de **Frameworks**, l'un des plus intelligents des programmes pour l'IBM serait sur le point de sortir une version de ce programme pour le Mac, utilisant toutes les ficelles des fenêtres, menus, etc. Si c'est vrai, ça va faire mal !

De son côté, **Microsoft** ne met pas tous ses œufs dans le même panier. D'un côté, il s'apprête à vendre **Excel**, porte-drapeau des programmes "anti-intégrés" de l'autre, il met la dernière main à **Productivity Software's Mouseworks** : un programme intégré comprenant traitement de texte, base de données, et tableur, qui serait au Mac ce qu'**Appleworks** est à l'Apple //. Il est vrai que l'équipe du créateur d'**Appleworks** - **Rupert Lissner** - a rejoint **Microsoft** au printemps dernier. Prix prévu pour ce programme : 295 dollars.

Intégré ou non ?

C'est donc le débat de cette rentrée. En clair : vaut-il mieux construire des programmes gigantesques et intégrés offrant toutes les fonctions imaginables (mais pas toujours à leur maximum), ou est-il préférable d'utiliser simultanément plusieurs programmes dont chacun est le meilleur dans son domaine, en les

faisant fonctionner ensemble avec le **Switcher** ? (Rappelons que le **Switcher**, développé chez **Apple** permet de partitionner la mémoire d'un Macintosh de 512Ko en y installant plusieurs programmes, qui peuvent communiquer entre eux).

Avec **Jazz**, **Lotus** a choisi la première option. Las ! Ses premiers utilisateurs commencent à s'aviser que la base de données est un peu "simplette", que le tableur ne comporte même pas la possibilité de lier entre elles des feuilles de calculs, que les "macros" (commandes automatisées) sont absentes.

Avec **Excel**, **Microsoft** a précisément choisi l'option inverse. Ce tableur où seuls les graphiques sont intégrés offre ce qu'on peut réaliser de mieux dans le domaine (et notamment ces fameuses "macros"). Pas de traitement de texte ? Peu importe. On utilise **Excel** et **Word** avec le **Switcher** (et bien sûr un Macintosh de 512Ko) et on ajoute au meilleur des tableurs, le meilleur des traitements de texte. Une fraction de seconde suffit à transférer des données ou des graphiques d'**Excel** dans **Word**.

Les programmes intégrés sont devenus de véritables monstres. Les lignes de programme sont si nombreuses, que l'utilisation de nombreuses disquettes s'impose de toutes façons. Le "debuggage" se fait problématique. Et, au bout du compte, les utilisateurs disposent de fonctions qu'ils n'utilisent pas forcément.

Le **Switcher** leur rend la liberté de choisir le meilleur programme de sa catégorie, et de l'associer avec un autre programme, lui aussi excellent. La possibilité d'étendre à 1 Méga-octet la mémoire vive du Macintosh donnera à ce véritable "pont inter-programmes" toute sa valeur. Or, précisément, le **Switcher**, qui en est déjà à sa version 3.9, se perfectionne. Il serait notamment capable désormais de faire fonctionner le Macintosh en "multitâches". En clair, même si on fait basculer le Macintosh sur un des programmes du **Switcher**, les autres programmes continuent

à fonctionner. On peut ainsi, par exemple commencer à écrire un texte, pendant que le tableur recalcule des formules. Si avec le Switcher, le Mac accède au rang d'ordinateur "multitâches", la vie des programmes intégrés risque de devenir décidément très difficile.

Adresses

Micro Magic
Millersville, MD USA

Com Log
Scottsdale, AZ

Manx Software Systems
Box 55, Shrewsbury, NJ 07701

Version Soft
66, rue Castagnary - 75015 Paris

Controle X
94, rue Lauriston - 75016 Paris

Ordinateur Express
3, rue Pelouze - 75008 Paris

Hayden
600 Suffolk Street Lowell,
MA 01854

**Personal Computer
Peripherals Corporation,**
6204 Benjamin Road Tampa
FL 33614

Paradise Systems
Tél. : (800) 5527 7977
en Californie

First Class Peripherals
3579 Highway 50 East,
Carson City NV 89701

Personics Corporation
2352 Main street, Concord
MA 01742

Hello Informatique
1, rue de Metz - 75010 Paris

EnterSet
410 Townend St, San Francisco,
CA 94107

Ann Arbor Softworks
308 1/2 S. State Street,
Ann Arbor MI 48104

Silicon Beach Software
PO Box 261430 San Diego,
CA 92126

ProVUE
222 22nd St, Huntington Beach,
CA 92648



Courrier des lecteurs

Olivier Herz

Je voudrais vous signaler un truc au sujet des masques ; il est annoncé dans Pom's qu'ils occupent 6 secteurs sur la disquette. En fait, on peut faire entrer un en 5 secteurs très facilement. Il suffisait de se souvenir (ces informations m'ont été fournies par le livre "Beneath Apple Dos") qu'un fichier requiert un secteur seulement pour la liste des pistes/secteurs. Or, les \$400 (1024) adresses de l'écran devraient tenir en 1024:4 = 4 secteurs, donc au total 5. D'où vient le secteur supplémentaire ? Il y a deux raisons : premièrement, un fichier binaire commence par 4 octets indiquant l'adresse de chargement et la longueur. En plus, un bug du Dos (signalé dans le livre précité) fait qu'il y a toujours un octet de trop écrit dans un fichier. Quels sont les remèdes pour éliminer ces 5 octets qui débordent ? On sait depuis le n° 1 qu'un écran texte est criblé de "trous" de huit octets, à chaque fin de groupes de 128 octets. La fin de l'écran coïncide justement avec ces octets superflus, c'est-à-dire que la vraie fin de l'image texte est à \$7F7 (dernier octet). Donc, la longueur utilisable est de \$3F8 et non \$400. Pour corriger Gesmask, il suffit de modifier la ligne 605 en remplaçant le paramètre "L1024" par "L\$3F8". Pour Gesmask Modifié, ce sera la ligne 609. Il faut noter que cette astuce marche aussi pour les images HGR : on peut les stocker en 33 secteurs avec : BSAVE nom, A\$2000 (ou 4000 pour HGR2), L\$FF8. Enfin, une autre astuce pour de beaux listings. En écrivant des REM, et en les

listant, on est forcé de voir imprimer les "REM" avec leur numéro de ligne. Il existe une astuce facilement accessible avec un lle ou c, qui consiste à insérer des caractères Del (ASCII 127), ce qui a pour effet sur la plupart des imprimantes (du moins sur Epson), d'effacer un caractère du buffer. Il faut donc écrire, juste après une REM, 6 caractères Del plus le nombre de Del correspondant au nombre de caractères qui forment le numéro de ligne, ni plus ni moins.

Une petite annonce : j'ai trouvé la solution complète du jeu d'aventure Dark Crystal, avec les plans et tous les détails. Si vous êtes coincés, écrivez-moi et je vous repêcherai. Sachez en tout cas que le gros bon sens règne dans ce jeu : les endroits intéressants sont tous marqués d'un indice particulier. Bonne chance !

Thierry HAN
Ambassade de Corée
B.P. 301,
Yaoundé CAMEROUN

Sans doute avez-vous remarqué qu'avec le DOS 3.3, lors de la lecture d'un fichier TEXT avec MON I actif, les minuscules apparaissent sous forme de caractères en inverse. Ce n'est pas bien grave, mais il me semble que l'on puisse y remédier. L'instruction 9FAA - STA \$AA5C semble inutile puisque l'accumulateur contient justement le caractère stocké en \$AA5C. Je

propose donc le patch suivant :

9FA7: EA NOP
9FA8: 09 80 ORA #\$80
9FAA: 20 C5 9F JSR \$9FC5

Yvan KÆNIG
Mosaïque GERBINO
4, avenue du Stade
06220 VALLAURIS

D'après le "Reference Manual" et bien d'autres écrits, les mémoires \$C000 à C00F sont équivalentes. Pourtant, en étudiant certains programmes professionnels (Print Shop, Dbmaster, par exemple), on rencontre des choses telles que :

STA \$C000

STA \$C00C

STA \$C00E etc...

Est-ce simplement une coquetterie ou y-a-t-il une raison ?

Bernard GIBERT
c/o Total-Indonésie
P.O. Box 6
BALIK PAPAN
EAST KALIMANTAN
INDONESIA

Il est exact que sur l'Apple II+, les adresses \$C000 à \$C00F (KEYBOARD) et \$C010 à \$C01F (KEYBOARD STROBE), sont identiques. Mais il n'en est plus du tout de même sur le //e et le //c. Vous trouverez ci-contre les adresses du //e telles qu'on les extrait à divers endroits du Manuel de Références.⇒

En raison de besoins très particuliers, j'ai été amené à écrire moi-même un programme de gestion de fichiers en accès direct. Il est écrit en Applesoft, sous DOS 3.3. J'utilise pour le tri la "routine de tri rapide en assembleur", parue dans le n°11 de Pom's. L'ensemble, très facilement adaptable, me donne entière satisfaction. Or, voici que j'ai un nouveau fichier à écrire qui atteindra, et même probablement, dépassera le Méga-octet. Tout naturellement, j'ai pensé à utiliser ProDOS. J'ai pu, sans problème, transférer le programme basic et l'adapter à ce nouvel environnement. Par

nom	état	adres	lec/écr	fonction
KEYBOARD	-	\$C000	*	lecture du clavier
80STORE	off	\$C000	*	utiliser RAMRD/RAMWRT
"	on	\$C001	*	accès aux pages d'affichage par PAGE2
RAMRD	off	\$C002	*	lire en mémoire principale
"	on	\$C003	*	lire en mémoire auxiliaire
RAMWRT	off	\$C004	*	écrire en mémoire principale
"	on	\$C005	*	écrire en mémoire auxiliaire
SLOT CXROM	off	\$C006	*	ROM interne en \$Cx00
"	on	\$C007	*	ROM d'un périphérique en \$Cx00
ALTZP	off	\$C008	*	pile, page 0 et carte langage en mémoire principale
"	on	\$C009	*	pile, page 0 et carte langage en mémoire auxiliaire
SLOT C3ROM	off	\$C00A	*	ROM interne en \$C300
"	on	\$C00B	*	ROM d'un périphérique en \$C300
80COL	off	\$C00C	*	affichage en 40 colonnes
"	on	\$C00D	*	affichage en 80 colonnes
ALTCHARSET	off	\$C00E	*	ensemble primaire de caractères
"	on	\$C00F	*	ensemble alternatif de caractères
KRDSTRB	-	\$C010	*	remise à 0 du clavier
"		\$C010	*	lire l'état de ALTCHARSET
"		\$C013	*	lire l'état de RAMRD
"		\$C014	*	lire l'état de RAMWRT
"		\$C015	*	lire l'état de SLOT CXROM
"		\$C016	*	lire l'état de ALTZP
"		\$C017	*	lire l'état de SLOT C3ROM
"		\$C018	*	lire l'état de 80STORE
"		\$C019	*	lire la suppression verticale (POM'S 17 page 54)
"		\$C01A	*	lire l'état de TEXT
"		\$C01B	*	lire l'état de MIXED
"		\$C01C	*	lire l'état de PAGE2
"		\$C01D	*	lire l'état de HIRES
"		\$C01F	*	lire l'état de 80COL
TEXT	off	\$C050	*	mode graphique
"	on	\$C051	*	mode texte
MIXED	off	\$C052	*	mode non-mixte
"	on	\$C053	*	mode mixte
PAGE2	off	\$C054	*	accès page 1 (si 80STORE est off) accès mémoire principale (si 80STORE est on)
"	on	\$C055	*	accès page 2 (si 80STORE est off) accès mémoire auxiliaire (si 80STORE est on)
HIRES	off	\$C056	*	accès basse résolution (si 80STORE est off) PAGE2 commute la page texte (si 80STORE est on)
"	on	\$C057	*	accès haute résolution (si 80STORE est off) PAGE2 commute aussi le hires (si 80STORE est on)

nouvel environnement. Par contre, je ne sais comment adapter la routine de tri. Pourriez-vous me conseiller ?

Georges DELARUE
2 Chemin des Chambons
38650 MONESTIER DE CLERMONT

Le programme TRITABLEAU.OBJ0 est compris entre les adresses \$9AA6 et \$9C3A ; c'est-à-dire en plein milieu des buffers du DOS. Par conséquent, il ne peut marcher tel quel sous ProDOS, et il faut le faire fonctionner à un autre endroit. Transférons, par exemple, en \$90A6-\$923A. Deux méthodes sont possibles :

- si vous avez l'assembleur ToolKit, changer la ligne 28 du fichier source en ORG \$90A6 et réassembler,
- sinon, faire les opérations suivantes :

BLOAD TRITABLEAU.OBJ0,
A\$90A6
90AC:90
90CE:90
90FB:92
911D:92
BSAVE
TRITABLEAU.OBJ0,A\$90A6,L
\$195

Il ne reste plus qu'à transférer TRITABLEAU.OBJ0 et éventuellement les programmes de démonstration en Applesoft sur une disquette ProDOS. Il faudra faire commencer tous les programmes Applesoft utilisant la routine tri par HIMEM:37030 (\$90A6=37030), afin de ne pas écraser la routine avec les variables des programmes Applesoft.

De nombreux lecteurs nous ont demandé une récapitulation des sujets abordés dans cette rubrique.

Nous vous proposons donc ci-contre une liste classée par mot-clef des principaux sujets abordés. Cette liste, non exhaustive, renvoie aux différents numéros de la revue ou aux recueils.

Nous vous laissons juges du caractère arbitraire du choix des mots servant de clef...

- 15 APA : Un bug corrigé
- 18 AppleMouse II
- 14 Applesoft, Moniteur et Carte langage
- 15 AppleWriter : Un truc
- 14 AppleWriter et Epson FX80/FT
- 10 AppleWriter //e (Patch pour code ASCII 0)
- R2 Astérisque en Pascal
- 19 Basicum et ProDOS
- 16 Beep amélioré
- 17 Calculs financiers
- 11 Calendrier perpétuel sur Visicalc
- 10 Carte mémoire Apple //e
- 13 Chat Mauve
- 12 Chat Mauve et Epson
- 10 Chat Mauve et PFS
- R2 Choose sur Visicalc
- 9 Club Apple ///
- R2 Clubs de Paris
- R1 Compacter les programmes
- 15 Comparaison de programmes Basic
- 14 Compatibilité Apple
- 10 Contrôle de la carte langage sur //e
- 16 Copya, Pascal et CP/M
- R1 Curseur en mode Flash
- 18 Date ProDOS
- 12 217 fichiers par disquette
- 14 Del sur Apple //e
- 17 DIF.OBJ : Un bug
- 17 Disk Manager 2
- 17 Disque virtuel
- 18 Disque virtuel 64 Ko
- 17 Edition de jeux
- 16 ê en Basic
- 15 ê sur Oki 84
- 11 Entrées/sorties : reprogrammation
- 19 Eprom d'Epson
- 18 Exec en Pascal
- 18 Gesmask et OKI 82A
- 17 Hard Copy 80 colonnes sur DMP

- 10 Hard Copy sur Seikosha
- 12 Hard Copy Texte
- 9 Hello Complet et page 3
- 13 Gescompte amélioré
- 9 Graphiques, quand tu nous tiens (Patch pour Epson MX-82IIIF/T)
- 16 IBM 36 vers Lisa
- 18 Icare : comment le charger
- 11 Init dans un Exec
- 16 Mac et ImageWriter
- R1 Mini-assembleur
- 10 Minuscules sur Seikosha GP 100
- 9 Moniteur étendu (Patch)
- 14 Multiplan et Centronics 152
- 14 Nouvelles touches du //e
- R1 Numéros de volume (neutraliser le contrôle)
- 10 Numéros de Volume : stockage sur la disquette
- 13 Matgraph (erratum)
- 12 Microcoupures, surchauffe...
- 14 Pascal 1.1, 1.2.
- 12 Peek et Poke : errata
- 11 Physique relativiste
- 10 PFS et Chat Mauve
- 9 Planche à voile !
- 11 Point ON ou OFF sur l'écran HGR ?
- 13 Poke 214,255
- R2 Poke 333,33 !
- 16 ProDOS et Eprom 2716
- 18 Programmation facilitée, en 80 colonnes
- R2 Programmation facilitée (à propos de...)
- 16 Purplesoft
- 19 Récupérer un écran 80 colonnes
- R1 Reset
- 15 Radiotélégraphie
- R2 Rom LC
- 15 Secteurs libres sur disquette
- R2 Splittage des programmes Basic
- 18 Startup et Date en Pascal
- 10 Tdump de fichiers TEXT sur écran ou imprimante
- 10 Tracé rapide de cercles en HGR
- 19 Transformer un tableau de chaînes en fichier binaire

pom's

Disquettes

HAIFA source	(cf. Pom's n° 5)	à 55,00 F
H-BASIC	(cf. Pom's n° 8)	à 150,00 F
MUSIC	(cf. Pom's n° 10)	à 80,00 F
DBSTAG	(cf. Pom's n° 11)	à 450,00 F
JEUX A	(cf. Pom's n° 12)	à 80,00 F
JEUX B	(cf. Pom's n° 12)	à 80,00 F
BASICIUM	(cf. Pom's n° 13)	à 150,00 F
E.P.E.	(cf. Pom's n° 15)	à 150,00 F
PASCAL	(cf. Pom's n° 15)	à 80,00 F
MAX (Moniteur étendu)	(cf. Pom's n° 18)	à 150,00 F
DOMINOS	(cf. Pom's n° 19)	à 80,00 F

Recueils

N°1, recueil des revues 1 à 4	à 140,00 F
Disquettes d'accompagnement 1 à 4	à 150,00 F
N°2, recueil des revues 5 à 8	à 140,00 F
Disquettes d'accompagnement 5 à 8	à 190,00 F

Anciens numéros

Revues 4 7 8	à 35,00 F
Revues 9 10 11 12 13 14 15 16 17 18 19 20	à 40,00 F
Disquettes Apple II, //e, //c 1/2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20	à 55,00 F
Disquettes Macintosh 14/15/16 groupées	à 150,00 F
17 18 19 20	à 80,00 F

Abonnements

Pour 6 numéros à partir du n°

Abonnement à la revue seule	à 200,00 F
Abonnement revue + disquettes Apple II, //e, //c	à 480,00 F
Abonnement revue + disquettes Macintosh	à 600,00 F

Total TTC :

Supplément avion hors CEE : 15,00F par numéro et/ou disquette :

Montant du règlement :

Envoyez ce bon et votre règlement à :

EDITIONS MEV, 64 rue des Chantiers 78000 VERSAILLES

Nom :

Adresse :

Applemania

Symptômes.

- | | OUI | NON | | OUI | NON |
|---|--------------------------|--------------------------|---|--------------------------|--------------------------|
| 1. Je veux connaître PRODOS sans être un pro du DOS | <input type="checkbox"/> | <input type="checkbox"/> | 5. Pianoter des nuits entières, c'est planant. Mais planer tout seul, c'est frustrant. | <input type="checkbox"/> | <input type="checkbox"/> |
| 2. Ami des bêtes, je cherche un chat à puces pour colorier ma souris en mauve | <input type="checkbox"/> | <input type="checkbox"/> | 6. Je veux relier mon lave-vaisselle à mon Apple II. | <input type="checkbox"/> | <input type="checkbox"/> |
| 3. Devenir à l'aise comme Blaise en Pascal, quel pied ! | <input type="checkbox"/> | <input type="checkbox"/> | 7. Appeler un S.O.S. informatique quand la technique coince, ça dépanne. | <input type="checkbox"/> | <input type="checkbox"/> |
| 4. Mon baud à 7 bits cherche un 8 ^e pour compléter son protocole X-On X-Off en vue d'une liaison durable | <input type="checkbox"/> | <input type="checkbox"/> | 8. Je veux apprendre à mieux gérer mon bureau en le jetant par les fenêtres de Macintosh. | <input type="checkbox"/> | <input type="checkbox"/> |

3 OUI et MOINS : vous êtes en pleine incubation, prenez des mesures !

4 à 7 OUI : vous êtes contagieux, vous serez bien entouré au Club Apple.

8 à 10 OUI : bravo, venez d'urgence cultiver votre passion au Club Apple.

Remède : REJOIGNEZ LE CLUB APPLE

Plein, plein d'avantages vous y attendent :

- une messagerie électronique gratuite ;
- un numéro de téléphone pour consulter les ingénieurs Apple 7 jours sur 7 ;
- tous les mois, le journal du Club : L'Echo des Apple ;
- des conférences, réunions d'information, projections privées et rencontres avec des "pros" ;
- des stages d'initiation et de perfectionnement animés par les "gourous" du Club ;
- des voyages organisés en Californie ;
- une librairie spéciale Club ;
- un Apple Check pour bénéficier de conditions de crédit préférentielles ;
- des rencontres organisées avec d'autres Applemaniaques.

BULLETIN-REPONSE

Je désire recevoir, sans engagement de ma part, une documentation complète sur le Club Apple et votre formulaire d'inscription.

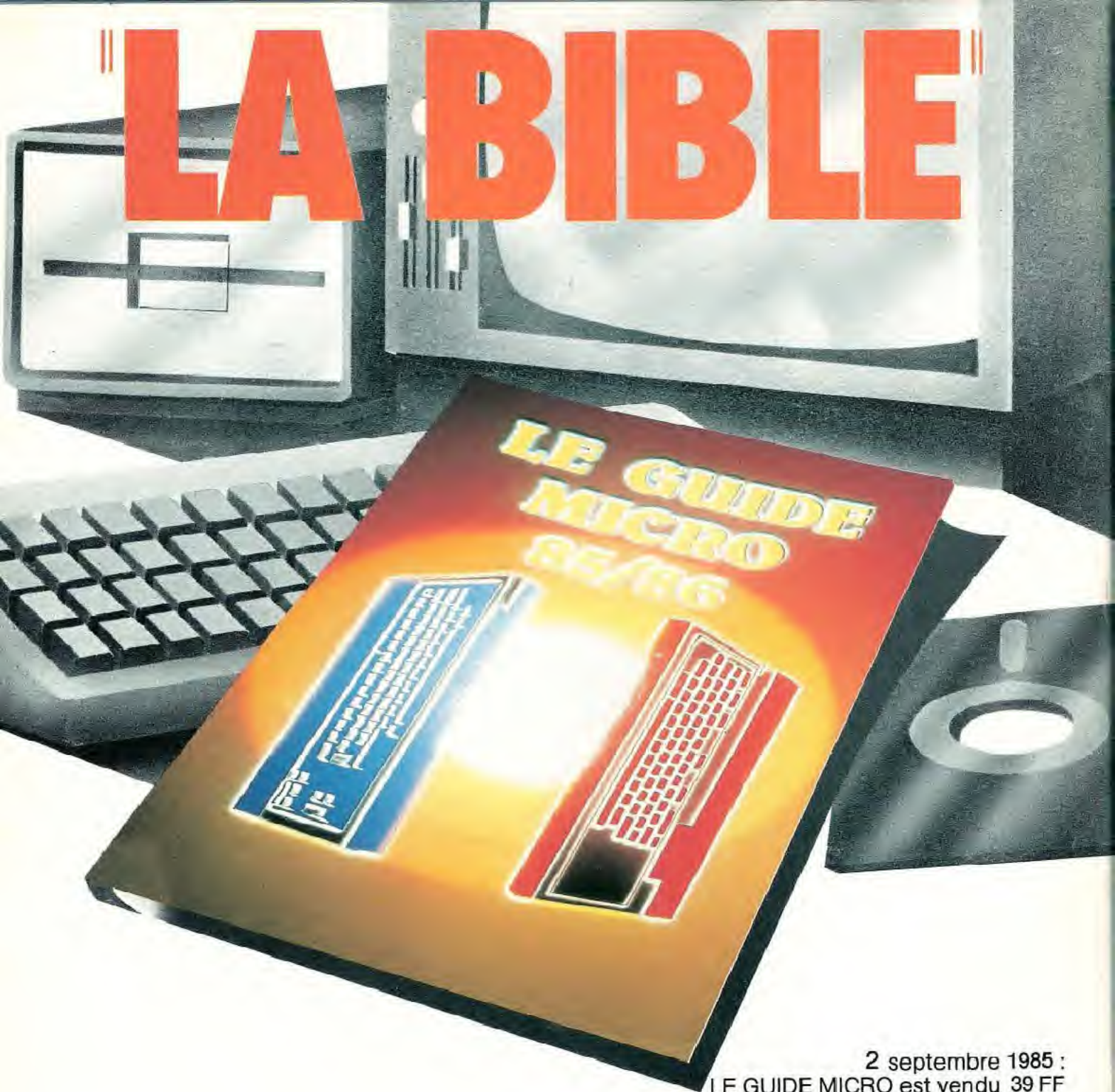
NOM _____

ADRESSE _____

CODE POSTAL _____



LA BIBLE



2 septembre 1985 :
LE GUIDE MICRO est vendu 39 FF
chez votre marchand de journaux.

216

FICHES TECHNIQUES détaillées des micro-ordinateurs du marché (de 400 à 60 000 FF). Le GUIDE MICRO vous fournit une foule de renseignements précieux.

88

BANCS D'ESSAI approfondis d'ordinateurs et de logiciels (gestion, utilitaires, jeux). Le GUIDE MICRO prend parti et dit lesquels acheter.

1

TABLEAU géant des 125 imprimantes valant moins de 15 000 FF. Le GUIDE MICRO vous aide à choisir en fonction de vos besoins.

3000

ADRESSES utiles : boutiques, clubs, constructeurs. Le GUIDE MICRO les répertorie par ordre alphabétique et par département.

Que vous soyez professionnel ou amateur, chevronné ou néophyte, le GUIDE MICRO est un « must ». Une aide décisive avant l'achat d'un micro ou d'une imprimante. Un outil de référence tout au long de l'année. Ne le manquez pas !

**LE GUIDE MICRO
DE
L'ORDINATEUR
L'INDIVIDUEL**